Implementation of an expert system for xenon spatial control in pressurized water reactors

Sun-Kyo Chung

Iowa State University

Follow this and additional works at: http://lib.dr.iastate.edu/rtd

Part of the Nuclear Engineering Commons

Recommended Citation
INFORMATION TO USERS

The most advanced technology has been used to photograph and reproduce this manuscript from the microfilm master. UMI films the original text directly from the copy submitted. Thus, some dissertation copies are in typewriter face, while others may be from a computer printer.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyrighted material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each oversize page is available as one exposure on a standard 35 mm slide or as a 17" × 23" black and white photographic print for an additional charge.

Photographs included in the original manuscript have been reproduced xerographically in this copy. 35 mm slides or 6" × 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
Implementation of an expert system for xenon spatial control in pressurized water reactors

Chung, Sun-Kyo, Ph.D.
Iowa State University, 1988
PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy. Problems encountered with this document have been identified here with a check mark √.

1. Glossy photographs or pages
2. Colored illustrations, paper or print
3. Photographs with dark background
4. Illustrations are poor copy
5. Pages with black marks, not original copy
6. Print shows through as there is text on both sides of page
7. Indistinct, broken or small print on several pages √
8. Print exceeds margin requirements
9. Tightly bound copy with print lost in spine
10. Computer printout pages with indistinct print
11. Page(s) lacking when material received, and not available from school or author.
12. Page(s) seem to be missing in numbering only as text follows.
13. Two pages numbered. Text follows.
14. Curling and wrinkled pages
15. Dissertation contains pages with print at a slant, filmed as received
16. Other:

________________________________________
________________________________________
________________________________________

UMI
Implementation of an expert system for xenon spatial control in pressurized water reactors

by

Sun-Kyo Chung

A Dissertation Submitted to the Graduate Faculty in Partial Fulfillment of the Requirements for the Degree of DOCTOR OF PHILOSOPHY

Major: Nuclear Engineering

Approved:

Signature was redacted for privacy.

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa
1988
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>1. INTRODUCTION</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1 Review of Xenon Instability in Thermal Reactors</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Background on Expert Systems</td>
<td>5</td>
</tr>
<tr>
<td>1.3 Aims of This Research</td>
<td>17</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2. XENON SPATIAL CONTROL METHODS</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Optimal Control Methods</td>
<td>23</td>
</tr>
<tr>
<td>2.1.1 Variational calculus method</td>
<td>23</td>
</tr>
<tr>
<td>2.1.2 Dynamic programming</td>
<td>24</td>
</tr>
<tr>
<td>2.1.3 Mathematical programming</td>
<td>25</td>
</tr>
<tr>
<td>2.1.4 Pontryagin-minimal time</td>
<td>27</td>
</tr>
<tr>
<td>2.1.5 Linear-quadratic-gaussian</td>
<td>28</td>
</tr>
<tr>
<td>2.2 Heuristic Method</td>
<td>29</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. ARCHITECTURE OF EXPERT SYSTEMS</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1 Introduction</td>
<td>32</td>
</tr>
<tr>
<td>3.2 Components of an Expert System</td>
<td>34</td>
</tr>
<tr>
<td>3.2.1 Knowledge base</td>
<td>34</td>
</tr>
<tr>
<td>3.2.1.1 Semantic networks</td>
<td>35</td>
</tr>
<tr>
<td>3.2.1.2 Production system</td>
<td>36</td>
</tr>
<tr>
<td>3.2.1.3 Frames</td>
<td>37</td>
</tr>
<tr>
<td>3.2.1.4 Logical expression</td>
<td>38</td>
</tr>
<tr>
<td>3.2.2 Inference Engine</td>
<td>40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. IMPLEMENTATION OF EXPERT SYSTEM ON MICROCOMPUTER</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1 Introduction</td>
<td>43</td>
</tr>
<tr>
<td>4.2 Microcomputer-based Expert System Development</td>
<td>43</td>
</tr>
<tr>
<td>4.3 Disadvantages and Advantages of Implementing Expert Systems on Microcomputers</td>
<td>45</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5. XSCD - AN EXPERT SYSTEM FOR XENON SPATIAL CONTROL</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1 Reactor Control Model for Load-Follow of a PWR</td>
<td>54</td>
</tr>
<tr>
<td>5.2 Knowledge Representation for Load-Follow of a PWR</td>
<td>72</td>
</tr>
<tr>
<td>5.2.1 Introduction</td>
<td>72</td>
</tr>
<tr>
<td>5.2.2 Core state representation and pattern-matching</td>
<td>78</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6. RULE LEARNING AND TESTING OF XSCD EXPERT SYSTEM</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1 Introduction</td>
<td>94</td>
</tr>
<tr>
<td>6.2 Selection of Weight Values for Control Variables</td>
<td>96</td>
</tr>
<tr>
<td>6.3 Determination of Unit Control Input Values</td>
<td>100</td>
</tr>
<tr>
<td>6.4 Testing of the XSCD Expert System</td>
<td>117</td>
</tr>
</tbody>
</table>
7. CONCLUSIONS AND RECOMMENDATIONS .......................... 143
   7.1 Conclusions ............................................. 143
   7.2 Recommendations for Future Work ......................... 148
      7.2.1 Implementation of XSCD for a load-follow
            of a PWR ........................................ 150
8. BIBLIOGRAPHY ................................................. 153
9. ACKNOWLEDGEMENTS ............................................ 160
10. APPENDIX A. SOURCE LIST OF STEADY.FOR PROGRAM ......... 161
11. APPENDIX B. EVALUATION OF MATRICES OF A, B, C, D, 
    AND E .................................................... 174
12. APPENDIX C. SOURCE LIST OF PREDICT.FOR PROGRAM ....... 183
13. APPENDIX D .................................................. 207
    13.1 Description for Implementing the XSCD Expert 
        System .............................................. 207
    13.2 Source List of the XSCD Expert System ................. 210
14. APPENDIX E. SOURCE LIST OF GRAPHIC PROGRAMS .......... 265
LIST OF TABLES

TABLE 1. Comparison of data processing and knowledge engineering ................. 14

TABLE 2. Error bands for each pattern of APOERR and AAOERR .................. 83

TABLE 3. All possible power level patterns and the corresponding control actions .......... 87

TABLE 4. All possible power shape patterns and the corresponding control actions .......... 90

TABLE 5. The weight values assigned to control variables .......................... 99

TABLE 6. Examples of desired load-follow patterns ............................. 105

TABLE 7. The unit control input values of control variables ....................... 111

TABLE 8. The best control inputs of control variables .......................... 113
LIST OF FIGURES

FIGURE 1. Structure of xenon spatial control expert system ........................................ 22
FIGURE 2. Subdivision of magnitude of power and AO error ........................................ 84
FIGURE 3. Combination of state variables for power level patterns ................................. 86
FIGURE 4. Combination of state variables for power shape patterns ............................... 89
FIGURE 5. Adaptation of control variables using forward-chaining inference ................... 102
FIGURE 6. Performance indexes versus DELT variation .................................................. 108
FIGURE 7. Performance indexes versus DELB variation .................................................. 109
FIGURE 8. An example rule in the XSCD expert system ................................................. 112
FIGURE 9. Reactor power and AO value as a function of time for load-follow pattern E .......... 114
FIGURE 10. The best control strategy corresponding to Figure 9 .................................... 115
FIGURE 11. Initial system state representation ............................................................... 118
FIGURE 12. Flux in units of $1.42\times10^{14}$/cm$^2$sec and xenon in units of $2.03\times10^{15}$/cm$^2$sec versus core height at time equal 4 hours and power equal 96 % of full power .................................................. 121
FIGURE 13. Flux in units of $1.42\times10^{14}$/cm$^2$sec and xenon in units of $2.03\times10^{15}$/cm$^2$sec versus core height at time equal 9 hours and power equal 80 % of full power .................................................. 122
FIGURE 14. Flux in units of $1.42\times10^{14}$/cm$^2$sec and xenon in units of $2.03\times10^{15}$/cm$^2$sec versus
core height at time equal 17 hours and power equal 88 % of full power . . . . . . . . . . . . . 123

FIGURE 15. Flux in units of $1.42 \times 10^{14}/\text{cm}^2\text{sec}$ and xenon in units of $2.03 \times 10^{15}/\text{cm}^3\text{sec}$ versus core height at time equal 18 hours and power equal 92 % of full power . . . . . . . . . . . . . 124

FIGURE 16. Flux in units of $1.42 \times 10^{14}/\text{cm}^2\text{sec}$ and xenon in units of $2.03 \times 10^{15}/\text{cm}^3\text{sec}$ versus core height at time equal 19 hours and power equal 96 % of full power . . . . . . . . . . . . . 125

FIGURE 17. Reactor power and AO value as a function of time for load-follow pattern A . . . . . . . . . . . . . 126

FIGURE 18. Integrated xenon concentration for load-follow pattern A . . . . . . . . . . . . . . . . . . . 127

FIGURE 19. The best control strategy corresponding to Figure 17 . . . . . . . . . . . . . . . . . . . 128

FIGURE 20. Reactor power and AO value as a function of time for load-follow pattern B . . . . . . . . . . . . . 132

FIGURE 21. The best control strategy corresponding to Figure 20 . . . . . . . . . . . . . . . . . . . 133

FIGURE 22. Reactor power and AO value as a function of time for load-follow pattern C . . . . . . . . . . . . . 135

FIGURE 23. The best control strategy corresponding to Figure 22 . . . . . . . . . . . . . . . . . . . 136

FIGURE 24. Reactor power and AO value as a function of time for load-follow pattern D . . . . . . . . . . . . . 138

FIGURE 25. The best control strategy corresponding to Figure 24 . . . . . . . . . . . . . . . . . . . 139
1. INTRODUCTION

1.1 Review of Xenon Instability in Thermal Reactors

It is well known that fission-product poisoning significantly affects the operation of a thermal reactor. When the reactor is first brought to power; the fission products start to accumulate in the fuel and build up to an equilibrium. Although there are many different kinds of fission products, the two that give rise to the most serious thermal reactor operational problem are Xe-135 and Sm-149, having large thermal neutron absorption cross sections. The buildup of these fission products introduces a negative reactivity effect that must be overcome by a positive reactivity insertion to maintain criticality. The sharp drop in boron concentration required to control the reactor at beginning of life (BOL) is mainly due to the buildup of these fission products.

Xe-135 is the most important fission product poison because it has an exceptionally large absorption cross section for thermal neutrons (approximately $2.4 \times 10^6$ barns). The isotope is formed, to a small extent, directly from the fission process of U-235. The fission yield for this process is 0.002. However, most Xe-135 comes from the radioactive decay of I-135. I-135 comes mostly from the
decay of Te-135 (fission yield 6.5 %). Since Te-135 decays rapidly to I-135 (half life is around 19.2 seconds), it is normally assumed that all I-135 is produced directly in the fission of U-235.

When the reactor is first brought to power, the Xe-135 concentration rises slowly to equilibrium. This is due primarily to the relatively long half-lives of I-135 and Xe-135. Operationally, as xenon builds up, other poisons in the core (control material such as boric acid in the coolant) must be removed, thus maintaining criticality. After approximately 40 hours of power operation, Xe-135 reaches equilibrium, a point at which the production of Xe-135 is equal to the removal of Xe-135 by neutron absorption and radioactive decay.

Sm-149 is a stable isotope with an absorption cross section of $5.3 \times 10^4$ barns for thermal neutrons. It is the end product of the mass 149 fission product group. Its predecessor is Pm-149. Since the half-lives of precursors of Pm-149 are short, it may be supposed that the latter is a direct product of fission. The process of the formation of Pm-149 and Sm-149 in a reactor is thus analogous to that considered for I-135 and Xe-135. Because Sm-149 is not radioactive, it is found that its equilibrium concentration during reactor operation is not flux level dependent;
rather, it depends on the design parameters of the reactor. Due to the longer time constant, smaller fractional fission yield, and smaller absorption cross section, Sm-149 does not play as large a part in the design and operation of power reactors as does Xe-135.

In a large reactor, operating at high thermal neutron flux levels at which the rate of xenon burnout is important relative to the rate of xenon decay, the phenomenon known as xenon instability can occur. This term refers to the process whereby a change in power density, whether on a local or global basis, perturbs the xenon and iodine distribution and the distribution of the thermal neutron absorption cross section oscillates in the core [1-3]. Introduction of a small tilt in the neutron flux distribution, for example, from top to bottom of the core, causes the xenon to burn out more rapidly at the top and less rapidly at the bottom. The rate of formation of I-135 at the top, the parent of the xenon, increases at the same time, but since the former has a half-life of 6.7 hours, there is a considerable time delay between the increase in the neutron flux and the associated increase in the rate of xenon formation in the top region of the core. Consequently, the net prompt result is that the xenon concentration decreases in the top region. The result is a
continued decrease in the xenon concentration and a steady increase in the thermal neutron flux, until the delayed production of xenon, by the decay of the increasing concentration of I-135, brings about an increase in the amount of Xe-135. Ultimately the growth of xenon from the iodine that is now forming more rapidly at the top region and the radioactive decay of xenon at the bottom region reverse the neutron flux distribution, and then the power peaks in the bottom region of the core. In due course, therefore, delayed production of xenon by the decay of I-135 will cause another reversal of the flux, and so on. Consequently, a continuous series of reactor power oscillations, having a period of about a day, will occur unless appropriate control actions are taken to dampen them.

The power oscillations arising from xenon are not a nuclear hazard, in the sense that there is a danger that the reactor will become supercritical. The main problem is that a local increase in the neutron flux means that fission heat is generated more rapidly than is expected. There is thus a possibility of fuel damage resulting from an excess power density. For this reason, minimizing axial peak local power densities during reactor operation while maintaining the necessary load change flexibility is a very important safety aspect of nuclear reactor plant control.
The reactor operator has special procedures to control the axial power distribution using control mechanisms (such as full- and part-length control rods) to minimize the effects of xenon redistribution and axial power redistribution during load-follow operations.

The corresponding radial power distribution is a function of core design features such as core geometry, burnup, control rod location, and burnable poison disbursement in the radial direction. Thus, the radial power peaks are relatively constant and bounded over core operating life.

1.2 Background on Expert Systems

Expert systems, which originated from research into the implementation of tasks on a computer that were difficult or impractical by conventional procedural programming, refer to computer programs that mainly contain a collection of rules of thumb and domain facts. In other words, an expert system is a computer program that mainly manipulates symbols rather than numbers as in a conventional program. The purpose of an expert system is to provide advice to the user on the interpretation of data, symptoms, etc., in a well defined domain of knowledge. The system is developed by embodying within a computer, the knowledge of an expert in a given
area. The two main parts of an expert system are the knowledge base and the inference engine. The knowledge base consists of rules of thumb and domain facts. The inference engine consists of reasoning or problem-solving strategies on how to use knowledge to make decisions [4-6]. The knowledge base thus provide the key to expert system performance, while the inference engine provides the mechanism for its use.

It is difficult to pinpoint an exact date for the start of research on what is commonly called Artificial Intelligence (AI). It might be noted that expert systems are considered to be a subset of AI. In the 1950s, PERCEPTRON [7] appeared with the neural net technique. It was a self-organizing automaton which can be thought of as a crude model of the retina in the human eye. However, it could be taught to match only a limited class of patterns as Minsky and Papert proved later [8]. In the 1960s, a new approach, using the notion of a heuristic search, was introduced by Ernst and Newell at Carnegie-Mellon University. Their work culminated in General Problem Solver (GPS) [9]. The generality of GPS was confined to a restricted domain of puzzles with a relatively small set of states and well-defined formal rules. That is, GPS did its work in a formalized micro-world where the problems are, in
human terms, not significant. In the 1970s, however, Lindsay and Buchanan et al. at Stanford University introduced an expert system, DENDRAL [10], which modeled the knowledge of a human expert to analyze mass spectrographic, nuclear magnetic resonance, and other chemical experimental data to infer the plausible structures of an unknown compound. At the same time, Martin and Fateman at MIT developed an expert system, MACSYMA [11], which performed differential and integral calculus symbolically. And so the real expert system, based on human expertise, was born, almost as a caricature of the real human expert. MYCIN [12], which diagnoses bacterial infections of the blood and prescribes suitable drug therapy, has become the hallmark of the expert system because it introduced several new features. Specifically, the knowledge base of MYCIN consists of hundreds of rules which incorporate certainty factors to allow the system to reach plausible conclusions from uncertain evidence. This is a very important feature because it makes it possible to arrive at correct conclusions even when some of the evidence is incomplete. Furthermore, MYCIN has the capability of explaining its own reasoning processes, for example, by responding to an inquiry as to why it was asked a particular question or how it reached a conclusion. Actually, it has been shown to
perform at a level equivalent to that of a human with years of training.

There are several other expert systems that have been developed: PROSPECTOR [13] which predicts mineral deposits from geological data, R1 [14] which is an order entry system configuring VAX computers, PSYCO [15] giving diagnosis of dyspepsia from patient histories, and so on.

As mentioned before, the main characteristic of an expert system is the knowledge base which consists largely of rules of thumb that have come to be called heuristics. There are several reasons for this definition [16-18]. Firstly, in practice, most of the difficult and interesting real-world problems do not have manageable algorithmic solutions as conventional procedural programming does, since they originate in a complex real-world, which generally resists precise description and rigorous analysis. Thus, the conventional symbolic and mathematical reasoning program has limited application to the area of complex real-world problems like diagnosing, predicting, planning, monitoring and controlling because it does not provide means for representing human expertise.

Secondly, humans who have the best expertise resulting from years, perhaps decades of practical experience achieve outstanding performance. Since expert systems embody and
use this human expert knowledge, they can also attain high
level of performance. This has proved to be true in the
several expert systems mentioned earlier.

The knowledge of the human expert is a scarce resource
whose refinement and reproduction creates wealth. Moreover,
the development of expertise in the human expert has
required education and internship years long. Extracting
human expertise from the human expert and putting it in the
knowledge base of an expert system can greatly reduce the
cost and the time of human expertise reproduction and
exploitation. In addition, we can easily update the
knowledge and speed up the process of knowledge refinement
because the knowledge resides in a computer program.

Hayes-Roth et al. [18] emphasized the significance of
knowledge in an expert system like this:

Machines that lack knowledge seem doomed to
perform intellectually trivial tasks. Those that
embody knowledge and apply it skillfully seem
capable of equaling or surpassing the best
performance of human experts. Knowledge provides
the power to do work; knowledge engineering is the
technology that promises to make knowledge a
valuable industrial commodity.

Really, knowledge from a human expert for a specific
technical domain is the main characteristic of the expert
system as well as the key ingredient of the expert system
performance.
A knowledge base is a type of database that contains facts or assertions and knowledge relationships about a certain subject. The knowledge base is thus unique to a particular domain, where domain refers to a specific field of knowledge.

The facts, sometimes called the working memory or temporary data store, consist of declarative knowledge about the particular problem being solved and the current state of affairs in the attempt to solve the problem [19]. So, facts are short-term information that can change rapidly during the course of consultation.

The knowledge relationship represents formulas showing the relationship among several pieces of information. The most common formula is the production rule proposed by Newell and Simon [20], which has a premise and a conclusion such as:

```
IF A and B, THEN C
```

Here "IF A and B" represents a premise and "THEN C" a conclusion. The premise expresses some condition on the state of the database, and whether or not it is satisfied at any given point. Thus the premise consists of patterns that
must match before the conclusion is reached. The conclusion specifies database changes to be made whenever the rule is satisfied and may be either an action to be performed or an implication that the system uses in further reasoning. When both conditions, "A and B", are present as facts or assertions, they match the left-hand side of the production rule, leading to the conclusion, it is "C". Therefore, the production rules having IF-THEN format can be easily used to represent rule of thumb knowledge.

Frequently, a rule has a series of conditions and it is the practice to make the first condition refer to the current context. That is, the first condition determines whether the rule as a whole is potentially true relative to the current state of the situation model.

The inference engine uses knowledge in the knowledge base to solve a specific problem by emulating the reasoning process of a human expert. The inference engine thus interprets the rules, monitoring the facts in the database and executing the conclusion of the rule that has its premise satisfied. The engine operates by scanning each rule's premise until one is found that can be successfully matched against the database. If no premise is true, no conclusion is drawn. Therefore, it is important for a set of rules to cover all possible situational combinations. Otherwise, the engine can not draw any conclusions.
The process of scanning each rule's premise is performed by the search strategy of the inference engine. If the number of rules is small, it is practical to search a random list. However, if the number of rules is large, there are problems in such a search. Specifically, it is possible that the length of the search may be finite but large enough to be practically infinite. To handle this kind of problem, to provide various kinds of search spaces and to use imprecise knowledge to search for a solution, AI research has developed several important inference mechanisms [21,22]: backward- and forward-chaining, depth-first and breadth-first search, heuristic search, etc.

We may have some uncertainty about the correctness of the conclusion that we get from the expert system. The credibility of the expert system can be greatly enhanced if it can explain to the user the reasons for a given conclusion. An expert system achieves this by retracing the chain of production rules that led to the conclusion. Then users can check the chain of reasoning of the expert system with the user's common sense knowledge about the system, noting the uncertainties associated with each step. Hence, this reasoning trace function, which explains how conclusions were reached and justifies the steps used to reach them, is very significant in an expert system for the following reasons [23]:
- Users can have more certainty in the results, more confidence in the system.
- It is easy for users to predict and test the effect of a change on the system operation.
- System development is faster since the system is easier to debug.
- The assumptions underlying the system's operation are made explicit rather than being implicit.

Actually, the ability of reasoning tracing and of explanation tracing, which is called "self-knowledge", has not only greatly enhanced the quality of expert system but also has been one of the valuable fruits of AI research.

Another easy way to define expert systems is to compare them with conventional programming. Basically, the difference is that expert systems manipulate knowledge while conventional programs manipulate data. Table 1, which is prepared by the Teknowledge company [24], a company devoted to engineering commercial expert systems, characterizes the difference between expert systems and conventional programs clearly.

The steps to be taken to build an expert system require building up the knowledge base from the simplest elements to the most complex, i.e., building up the concepts first, then rules, then models and then strategies. This is done by
TABLE 1. Comparison of data processing and knowledge engineering

<table>
<thead>
<tr>
<th>Data Processing</th>
<th>Knowledge Engineering</th>
</tr>
</thead>
<tbody>
<tr>
<td>Representation and use of data</td>
<td>Representation and use of knowledge</td>
</tr>
<tr>
<td>Algorithmic</td>
<td>Heuristic (Rules of thumb)</td>
</tr>
<tr>
<td>Repetitive process</td>
<td>Inferential process</td>
</tr>
<tr>
<td>Effective manipulation of large database</td>
<td>Effective manipulation of knowledge bases</td>
</tr>
</tbody>
</table>

The collaboration of a domain expert, who is a practicing expert in some technical domain, with a knowledge engineer, an AI specialist with skill in analyzing an expert's problem-solving process [25]. In the initial stage, the domain expert considers a test case to set up the desired system behavior for a range of typical problems. The knowledge engineer uses the test case to build up an initial set of rules, then to set up the overall model, and then to make a prototype expert system for a typical problem. When this prototype is implemented and the domain expert has approved it, the knowledge engineer and the domain expert can start to increase the domain of the problems, primarily with the development of rules with increased details and scope, so that continuing progress results in better system coverage.
of problems as well as better performance of the system in the given domain.

The construction and the demonstration of the preliminary prototype system having a limited domain are very important steps to get an early idea how the system might look when completed. If the implementation of the prototype is satisfactory, the system can be gradually expanded with more and more knowledge acquisition and refined to handle more in-depth domain problems. However, if it is not satisfactory, the prototype can be improved through continuous testing, debugging, and refining until the system is satisfactory.

Several distinctive features of an expert system can be identified, based on the characteristics of the knowledge base and the inference engine mentioned above. These include:

- Conclusions or decisions for a specific domain are more reliable and consistent since the expert system refers to a computer program, which has better consistency than the human.
- The domain of the expert system is limited to a specific domain of expertise.
- The system has the ability to explain its train of reasoning for the decision or conclusions. That
is, the self-knowledge function of an expert system is a significant trait.

- Expert systems can be more easily expanded than conventional programs since they are designed to grow incrementally. Therefore, the systems can be gradually improved as the problem domain develops.
- Expert systems can function with incomplete or uncertain data because of the characteristics of the inference engine and through the use of confidence or certainty factors.
- Facts and the inference mechanism in an expert system are clearly separated.
- The expert system is typically a heuristic rule-based system.
- The expert system may be especially useful as a consultant since it delivers advice as its output and has its self-knowledge abilities.
- The expert system can serve as a good trainer since it can show an example of good strategy in approaching a problem.

There are several areas where expert systems can be applicable as shown by Hayes-Roth et al. [18]. Also, Rychener [16] showed some expert systems which have proven successful.
1.3 Aims of This Research

As the nuclear power generation percentage of the utility power system increases, the load following capabilities of the nuclear power plant become more important. Since the nuclear units are an integral part of the utility power system, they must meet the operating requirements of the system for load following capability on a daily and an hourly basis.

Load following on the one hand, and certain equipment tests such as a turbine control valve test on the other represent two types of operations that might be done in a nuclear power plant. The former requires smooth power level changes according to an expected and predictable change in daily load demand and the latter represents a short duration perturbation. Both require appropriate axial power distribution control strategies to prevent an axial power oscillation, which as previously described appears because of the axial imbalance of xenon distribution following any power perturbation in a thermal reactor. Hence, the control of axial xenon oscillation has been an important research topic for some time.

Most commercial pressurized water reactors (PWRs) are sensitive to axial xenon oscillation in any power maneuvering situation during a given cycle of reactor
operation. A number of studies of this instability have been performed, specifically using optimal control techniques which have a quadratic objective functional [26-33]. Few, however, have been applied to actual reactor power maneuvering situations, since the optimal control techniques require a detailed mathematical model for a reactor's dynamic behavior which is practically either incomplete or unknown. Furthermore, the application of an optimal control technique to the actual power maneuvering situation, if it could be applied with some amount of effort, would require redesign or modification of the plant control, instrumentation and computer systems. Because of these reasons, currently, PWR plants generally use heuristic procedures employed in practical day-to-day operations to prevent the oscillations.

The basic idea behind the heuristic procedure is the Constant Axial Offset Control (CAOC) strategy [34,35]. The CAOC strategy is to keep constant axial power shape; it is performed by reactor operators. The operator must keep an Axial Offset (AO) value within a target AO band. AO is defined as

\[ AO = \frac{(PT - PB)}{(PT + PB)} \]

where PT and PB represent the fraction of full-rated power generated in the top and bottom halves of the core,
respectively. Thus, an AO value simply represents the normalized difference between the power in the top half of the core and the power in the bottom half of the core. These half core powers are measured by suitably placed neutron detectors located outside the reactor vessel.

Effective use of the CAOC procedure by the operator is in fact essential to meet technical specifications and operating guidelines, which place restrictions on the allowable AO band.

Humans, however, have the disadvantage that their reliability and consistency are much less suited to such a complex reactor control strategy than those for example of a computer. Therefore, it is very desirable to develop a computer-aided advisory system for the effective utilization of the CAOC procedure so that it can achieve improved operational reliability and consistency as well as reduction in the reactor operator's load.

Effective control of axial xenon oscillations using heuristic CAOC procedures is a knowledge- and experience-oriented task. The control action is normally done by operators who have several years of practical experience and knowledge of this control method. That is, the control action should be taken by operational experts.
Expert systems have the potential for providing assistance to reactor operators in order to achieve expert-level control actions preventing xenon oscillations under actual power maneuvering situations.

The purpose of this research is to develop a microcomputer-based advisory expert system which will help reactor operators take expert level control actions for the control of xenon axial oscillations in load following operation of a PWR. The goal of the research is thus not related to the stability of the reactor itself, but rather to the development of an expert system to provide advice on the best control strategy.

INSIGHT2+ [36], which is a problem-independent tool for building expert systems on a microprocessor-based computer system, was used to develop the expert system. Basically, the system utilizes a forward-chaining or data-driven inference engine and stores the well-known expert knowledge for the xenon oscillations, based on production rules [20], in its knowledge base. Since the xenon transient is due to changes in reactor power level and xenon oscillations result from an imbalance between the spatial distribution of the power density and xenon concentration, power level and power shape patterns are matched and then control adjustments of control variables are made based on patterns heuristically
selected in the knowledge base. To implement the expert system, a microcomputer-based simulation core control code of a PWR was developed using a model based on one-group diffusion theory with moderator temperature and density feedbacks [37]. In order to determine the best control actions, which maintain the desired power level as well as core axial offset within a band around a target value, the core control model iteratively predicts six different core state variables: relative error and its magnitude in reactor power level, relative error and its magnitude in AO value, and time rate of change of the power level and AO value.

In order to see the system performance more effectively, graphic routines were developed which are incorporated into the knowledge base. Figure 1 shows the components of the axial xenon oscillation control expert system, which is called Xenon Spatial Control Doctor (XSCD). The XSCD expert system is interfaced with a microcomputer-based simulation core control code of a PWR, graphic engine, and disk data base. The characteristics of each component is described in Chapter 5.
FIGURE 1. Structure of xenon spatial control expert system

(a) : represents passing of core state variables.
(b) : represents passing of control action.
(c) : represents activation of graphic program.
(d) : represents passing of graphic output to the CRT.
(e) : represents reading of input data.
(f) : represents writing of data for graphic programs.
(g) : represents reading of data for graphics.
2. XENON SPATIAL CONTROL METHODS

2.1 Optimal Control Methods

A number of optimal control methods, based on a quadratic objective functional (or performance index), have been applied during reactor power maneuvering situations. The fundamental goal of optimal control methods is to determine the control that minimize the quadratic objective functional subject to the constraints on the state and control variables.

Various approaches of optimal control methods to spatial power control can be classified, according to the optimization method used, under one of the following five general categories [38-41]: variational calculus method, dynamic programming, mathematical programming, Pontryagin-minimal time, and linear-quadratic-gaussian. Each optimal control method is briefly reviewed and compared in the following sections.

2.1.1 Variational calculus method

The variational calculus technique [42] is a general method for solving nonlinear dynamic optimization problems. It can be easily applied when continuous variables with equality constraints are used. The solution can be obtained
by multiplying the state equations by Lagrange multipliers and adding the state equations to the performance index to form the Hamiltonian of the problem. The resulting objective functional is required to be stationary with respect to the continuous state variables. The optimal control is obtained as a function of the state variables and Lagrangian multipliers.

Hanke [43] applied variational calculus methods to a slow xenon transient control problem for a one-dimensional core using spatially continuous control. The main theoretical disadvantages of this method are that discontinuous variables and inequality constraints are difficult to treat. As a result, a realistic description of control rods is difficult.

2.1.2 Dynamic programming

The purpose of dynamic programming [44] is to maximize the computational efficiency when solving complicated optimization problems. The standard procedure is to discretize all the variables and determine an optimal value of the objective function in a feedback form. This method can treat nonlinear state equations, different types of constraints, and nonanalytical objective functionals without generating any theoretical problems.
Stacey [45] applied this method to a three-dimensional nodal PWR core model. The objective was to minimize a measure of the power peaking during the transient. The optimal solution proceeded by calculating the xenon transient forward in time and performing the optimization backward in time. Neef [46] also used this method for optimizing control of a xenon transient in the same manner as Stacey [45].

Dynamic programming is a useful method for the xenon spatial control problem as it allow the treatment of nonlinearities and constraints as well as being fairly computationally efficient. However, this method requires excessive computer storage as the number of control and state variables increases in the problem. Thus, more advanced algorithms are needed before a practical on-line application to the power maneuvering situation is possible.

2.1.3 Mathematical programming

The basic idea of mathematical programming is to linearize either the core model or the objective functional and constraints over the entire maneuver and apply either linear or quadratic programming to obtain the optimal control solution [47-51]. Love [48] studied a one-dimensional load-follow control problem with iterative use
of a linearized and nonlinear nodal core model with thermal feedbacks. The objective functional consisted of minimizing the difference between the actual and desired power levels. The iterative solution starts with the nonlinear model to calculate the input parameters to the linear model. Using the input parameters, this linear model in turn calculates a new control by linear programming. The new control is again used to input the nonlinear model, etc., and the iterations are continued until the axial offset constraint is satisfied.

A noniterative quadratic programming approach [49-51] was used to solve the optimization problem. The control model was a quasi-stationary linearized nodal approximation in the sense that level changes were calculated through a nonlinear point reactor dynamics equation on the basis of the known load demand, and linear parameters were precalculated by a nonlinear node core model. The objective functional measured the deviation of the normalized power, xenon, and iodine distributions from the desired ones. Hard constraints can be imposed on total power, controller speeds and operating ranges, and axial offsets, etc.

This method, using linear and quadratic programming algorithms, is computationally efficient and could be used on-line as long as the number of nodes and time steps are
kept relatively small. The disadvantages of the method are that linearized models have to be used in developing the models and that the optimal control solution obtained is not in feedback form. That is, the solution is obtained in open loop form.

2.1.4 Pontryagin-minimal time

The basic idea of this method is to control the xenon and iodine imbalances causing power oscillations, instead of only forcing the power distribution to the desired shape. Since the objective is to dampen the xenon oscillations out as quickly as possible, the objective functional is just $\|l\|$ [29-31]. The core model is linearized using a perturbation approximation and the spatial dependence is described by assuming a two-term modal expansion. In particular, the flux dependence is decoupled from the core model equations leaving only two coupled ordinary differential equations involving the xenon and iodine concentrations as state variables. The equations describe the oscillating xenon and iodine distributions whose amplitude can be written as a complex variable. The optimal control is of the bang-bang type where the phase is divided into two regions and the control switches from one control limit to another when the sign of the switching function
changes. Using the method in practice requires an on-line means for tracking the xenon and iodine deviations in a phase plane and construction of switching function curves.

The disadvantage of the method is that the bang-bang control tends to produce undesirably large changes in the local power density as the control rods move from one control limit to another. The linearity assumption of this method makes it difficult to apply to a power maneuvering situation.

2.1.5 Linear-quadratic-gaussian

If the state equation is linear and the objective function quadratic in the state variables and not time integrated, and no constraints are used, the optimal control can be obtained in a state variable feedback form [52-55]. Blomsnes et al. [52] applied a state variable feedback method to control axial xenon oscillations in a model of a large PWR core. A feedback controller with adaptive and learning capability has been developed [53,54]. The basic method with no constraint has been modified by imposing hard constraints on the state and control variables [55].

An application of the linear-quadratic-gaussian theory to a power maneuvering problem has been pursued [56,57]. The core model is based on a linearization of the nodal slow
transient equations taking thermal feedback also into account. The nonmeasurable state variables, the xenon and iodine distributions, are estimated with a Kalman filter. Because of linearization, most of the methods are restricted to xenon oscillation control problems.

2.2 Heuristic Method

A heuristic method is not based on any formal optimization technique but on intuitive principles. Christie and Poncelet [38] showed a good example of a heuristic control method for the control of spatial xenon oscillations. Several practical, manually executed heuristic control procedures both in simulations and power reactor tests were systematically compared [59]. They all have as a goal the minimization of the difference between the actual power distribution and some target value. Controls are intuitively manipulated to achieve the goal. Bauer [60] has developed anticipatory control strategies preconditioning the core before load changes to avoid xenon oscillations after the power level changes. These strategies are based on the xenon and iodine distributions and their changes during power maneuvering.

The heuristic control strategy of maintaining the axial offset constant in load-follow operation leads to fairly
simple operational control procedures. The applicability of this control strategy to load-follow operation has been demonstrated in power reactor tests by Sipush et al. [35].

The feasibility of load-follow operation employing CAOC strategy with and without the use of part-length control rods (PLCRs) was decisively demonstrated by the control strategy. If PLCRs can be used, the control strategy allows load changes with fairly high ramp rates. However, without PLCRs, only slow load changes can be followed. Actually, this control strategy employs half-cycle damping using the proper method and timing of control rod insertion. By inserting control rods before the oscillation reaches its positive peak, then withdrawing the control rods at a later time an inverse oscillation can be set up to effectively cancel out the original xenon oscillation, leaving a relatively stable AO. However, this damping may not be economically feasible near the end of a fuel cycle when large quantities of water are necessary to achieve the boron removal required to compensate for control rod insertion.

Even when soluble boron levels are high, the insertion of control rods to dampen oscillations, with compensation by boron, generates waste water which is costly to recycle for cleanup.
The disadvantage of the heuristic method is that other parameters of interest in the goal are not optimized, which may lead to a loss of maneuverability. In other words, the parameters chosen heuristically may not cover all types of load-following operations.
3. ARCHITECTURE OF EXPERT SYSTEMS

3.1 Introduction

As stated in Chapter 1, an expert system is a computer program which can store a human expert's knowledge in a specific domain and use this knowledge to solve problems from the domain in an intelligent way [18].

Expert systems are based primarily on symbolic processing and declarative programming. Typical operations in an expert systems are list manipulation, matching and resolution. Newell and Simon [61] assumed that symbol processing is a necessary and sufficient condition for a computer to perform a general intelligent action. Declarative programming means that the objects and the conclusion that is defined by their associated rules are given by the static description of the objects themselves and are independent of other objects and of the program execution. Hence, declarative programming, supported by an AI programming language (LISP or PROLOG), supports the construction of a modular and maintainable expert system [62]. In addition, expert systems are usually embedded in an interactive programming environment because AI programs, which may be slow when used in an interpretative mode, can automatically be transformed or compiled into fast procedural programs.
Application of expert systems to difficult, ill-structured operational problems of a power plant is a promising approach because the solution of the problems requires domain knowledge, significant expertise, much experience, and rules of thumb. Furthermore, the expert system is very reliable. Although a human expert's advice for the problem domain is reliable, his advice may suffer because the expert is a human. Availability is another advantage of expert systems. Human expertise is the result of a long time of practical experience in a problem domain. However, after a computerized expert system is completed, we can use the system whenever needed, and also get more use from the expert by simply copying the computer program from one machine to another. On the other hand, computing speed and memory limitations are two of the practical problems that AI programs have. However, this problem can be successfully solved using the following techniques [63]:

- The originally, purely declarative, AI languages are enhanced with procedural features and even mixed with conventional languages (C, PASCAL and FORTRAN) in one integrated system.
- AI programs are automatically transformed or compiled into fast procedural programs.
• Conventional processors and memories are becoming cheaper and more powerful.

• New computer architecture is being developed that should be more suited to symbolic and logic processing [64].

In designing an expert system that will perform with the characteristics mentioned above, a more detailed review of components of the expert system will be needed. The knowledge base and inference engine, which are the main components of an expert system, are reviewed in more detail in the following sections.

3.2 Components of an Expert System

3.2.1 Knowledge base

A knowledge base contains facts and rules for a specific domain. Facts are information that can change rapidly during the course of a consultation and rules are procedures for analyzing facts and concluding new facts. The information in the knowledge base is not a passive collection of records, but a symbolic representation of a human expert's rules of thumb (or knowledge) in a specific domain for use by the inference engine. Thus, the knowledge base which contain facts and rules is used to determines new
facts based on what is already known. The knowledge base should be easy to develop and readily expandable.

Knowledge representation refers to the task of modelling the knowledge in terms of computer data structures. Selecting an appropriate knowledge representation scheme is very important because AI research has shown that the inference strategies and controls heavily depend on the knowledge representation scheme [65]. The knowledge representation scheme can effectively limit what the expert system can perceive, know, or understand [66].

The basic knowledge representation techniques are [6,21,67]: semantic networks, production systems, frames, and logical expressions. Each method is briefly reviewed as follows.

3.2.1.1 Semantic networks The semantic network is the oldest and the most general knowledge representational scheme. A semantic network consists of nodes, representing objects, concepts, and events, and links between the nodes, representing their interrelations. Consider, for example, the simple net:
where BIRD, PARROT, and WINGS are nodes representing objects or concepts and IS-A and HAS-A are the names of the link specifying their relationship. One feature of the semantic net is that relevant facts about an object can be inferred from the nodes to which they are directly linked, without a search through a large database. Flexibility is another feature of this scheme. New nodes and links can be defined as needed.

3.2.1.2 Production system The basic idea of the production system is that the data base consists of rules, called productions, in the form of IF-THEN condition-action pairs, in which the condition specifies some pattern. For example, a simple rule is

```
IF lamp A is red, and
lamp B is red, and
flow rate of waste water is greater than
the red line

THEN turn off the electric power of the
 evaporator, and
check the air gauge of the valve A, or
check the air gauge of the valve B.
```
Technically, the IF part is called a premise and each of the premise is called an "if-clause". The premise can be made up of several if-clauses by connecting with the logical operators "AND" or "OR". The THEN part is called the conclusion and can also contain one or more clauses.

Since an individual rule represents an independent piece of knowledge for a domain, the production system is a modular knowledge representation scheme. The modularity is one key feature of the production system because it facilitates human understanding and modification of systems with large amount of knowledge. Thus, this scheme is finding increasing popularity in large expert systems.

3.2.1.3 Frames A frame is the most recently developed knowledge representation scheme. A frame is a data structure that includes declarative and procedural information about an object. Thus a frame has knowledge "hooks" or "slots" for all of the information associated with the object. Each object consists of a set of slots. Some slots contain properties associated with the object of the frame. Other slots may include default values, sets of rules, pointers to other frames, or procedures.

Declarative representations are more easily understood and maintained due to their modularity. On the other hand, procedural representations are more efficient to use but
harder to maintain. The ability to use both declarative and procedural representation is a key feature of frames and makes the frames powerful, generic, and popular as more complex systems are being built.

3.2.1.4 Logical expression  A logical expression is based on propositional logic and predicate calculus, sometimes called predicate logic. Propositions are statements that are either true or false. Using compound statements of AND, OR, NOT, IMPLIES, and EQUIVALENT, propositions are linked together. Propositional logic is concerned with the truthfulness of compound statements according to rules for propagating the truthfulness of statements. For example, if X and Y are any two propositions,

\[
\begin{align*}
X \text{ AND } Y \text{ is true if } X \text{ is true and } Y \text{ is true;} \\
\text{otherwise } X \text{ and } Y \text{ is false.} \\
X \text{ OR } Y \text{ is true if either } X \text{ is true or } Y \text{ is true} \\
\text{or both.}
\end{align*}
\]

Statements about objects, both by themselves and in relation to other objects, are called predicates. A
predicate is applied to a specific number of objects and has a value of either TRUE or FALSE. An example of a predicate of one object is the predicate "is-red". For example, "is-red(apple)" is an assertion that says that an apple is red. This assertion is either true or false. Predicates can have more than one object. For example, the statement "is-greater-than(value A, value B)" is addressed by two objects with one predicate. The basic form of a predicate in predicate logic is

\[(\text{predicate name})[\text{object list}].\]

Following is an example of the logic expression;

\[
\begin{align*}
is\text{-}\text{greater}\text{-}\text{than}(\text{suction pressure, pump A}) \\
is\text{-}\text{greater}\text{-}\text{than}(\text{discharge pressure, pump B}) \\
is\text{-}\text{red}(\text{lamp signal of pump A}) \\
is\text{-}\text{red}(\text{lamp signal of pump B})
\end{align*}
\]

Propositional logic alone is not useful for the purpose of AI. Predicate logic, however, can express true or false propositions, speak of objects, postulate relationships between objects, and generalize the relationships over classes of objects. Thus, logical expression is a form of
predicate logic. Predicate logic has been brought to prominence by the popularity of PROLOG, a language which provides a flexible symbolic representation scheme that allows rules and facts to be coded in essentially the same way.

3.2.2 Inference Engine

The inference engine uses facts and rules from the knowledge base to solve a specific problem by emulating the reasoning process of a human expert. The AI approach for solving a problem consists of searching for a solution from the set of all possible solutions. This is known as the search space in AI terminology. Thus, the inference engine essentially consists of problem-solving strategies that use knowledge from the knowledge base to search for a solution.

There are two overall reasoning strategies used to search for a solution using a search space: forward-chaining (or data-driven) and backward-chaining (or goal-driven) inference mechanisms.

In forward-chaining, the reasoning proceeds from data or symptoms to hypotheses goal, i.e., given data, preconditions for the truth of certain hypotheses are tested. In other word, the reasoning proceeds bottom-up from the symptom or evidence toward conclusion implied by
the inputs. This process repeats until one or more top-level goals are established.

As described by Harmom and King [21], a forward-chaining system makes clear the distinction between the knowledge base and the working memory. The premises of the rules in the knowledge base are compared to the contents of working memory which contains fact. When a rule succeeds, its conclusion is placed in the working memory. Also, reasoning is described as a "match-act" cycle. First, the rules that can succeed, given the contents of working memory, are matched. One rule is selected and then the action is asserted into working memory. The system then proceeds to the next cycle and checks again to see what rules succeed. As a result, forward-chaining is easy to computerize and is suitable in planning, monitoring, and other types of expert systems which carry out tasks. Typical examples of forward-chaining systems are R1 [14] for configuring VAX computers and PDS [68] for monitoring turbines.

In backward-chaining, the reasoning proceeds from hypothesis (goal) to data or evidence, i.e., the inference engine first selects a goal to be tested and then seeks the evidence required to test the hypothesis. If a certain hypothesis turns out to be false, the system can cancel all
conclusions that preceded or followed the false hypothesis. In other words, to establish a goal, each of its premises is checked. Each of those premises then becomes a subgoal, and this process is applied recursively until a value is established for the top goal. If the possible outcomes are known and if they are reasonably small in number, backward-chaining inference is very efficient. MYCIN [12] is a well known example that uses a backward-chaining inference mechanism.
4. IMPLEMENTATION OF EXPERT SYSTEM ON MICROCOMPUTER

4.1 Introduction

As described in Chapter 1, a number of expert systems have been developed in a wide range of technical fields. Until recently, however, most of the expert systems were designed to be implemented on minicomputers or special workstations, making the expert systems available only to those few programmers who had access to such computer hardware. In other words, originally expert systems could only operate on expensive or special-purpose hardware such as symbolic processors. For example, Automated Reasoning Tool (ART) [69] runs on the LISP processor or DEC VAX series computers which are in the price range of $60,000 to $80,000.

When the AI market was still forming, a microcomputer was considered to have limited capability as a symbolic processor and therefore most expert system tools for microcomputer were written in an interpreted version of LISP. With the increasing popularity of knowledge-based expert systems, there has been a corresponding increase in the demand to apply expert system technology in a wide range of computing environments. As a result, expert systems have migrated from expensive or special-purpose hardware to low-
cost microcomputers and various efforts have been made to implement expert systems on microcomputers. Certain tools for knowledge-based expert system development can be taken out of the LISP or PROLOG environment in which they were originally developed and moved into the more standardized and structured programming environment needed to make efficient use of the limited capabilities of microcomputers. For example, Knowledge Engineering System (KES) [70] was first programmed on the microcomputer in LISP. Realizing that it was necessary to increase the KES speed and portability, the KES was ported to a C programming environment.

Microcomputers, as originally designed, had several limitations such as processing speed, data storage, and the 640K barrier. However, a number of enhancements of microcomputers have been made. The appearance of the Winchester disk dramatically enhanced the data storage capabilities of microcomputers. The new OS/2 operating system [71] is going to break the 640K barrier. In addition, the Intel 80386 microprocessor chip, which features thirty-two-bit architectures, is available for the microcomputers (thirty-two bits is the same size as the VAX computer). The microcomputer with the thirty-two-bit architectures can effectively and efficiently support
multiple megabytes of core storage and fifty or more megabytes of disk capacity. As a result, the difference between microcomputers and minicomputers is becoming increasingly blurred.

This chapter reviews several knowledge-based expert system development tools which are commercially available for microcomputers and briefly discusses the disadvantages and the advantages of implementing expert systems on microcomputers.

4.2 Microcomputer-based Expert System Development Tools

A number of different tools have been designed for implementing expert systems on microprocessor-based computer systems. They all are domain independent tools for building expert systems and provide low-cost entry into expert system applications. All are currently being used to develop expert systems in a wide range of technical fields and are undergoing enhancements for future releases. They utilize a variety of inference mechanisms and knowledge representation schemes. Particularly, they run on a variety of microcomputers. Some tools can run in 512K or less memory. However, the size of the knowledge base depends on the memory size; with more memory, larger knowledge bases are possible, although the maximum memory barrier of current microcomputers is 640K.
There are several expert system development tools which are commercially available for microcomputers. Based on the popularity of the systems, a few tools are briefly reviewed here.

INSIGHT2+ [36], the tool applied in this research, is an advanced development and runtime environment for expert systems. The Production Rule Language (PRL), based on the production rule, is used to make a knowledge base. Forward- or backward-chaining inference mechanisms can be utilized. It makes use of a compiler for knowledge bases. At the compile step, the knowledge base is reduced to a symbolic representation that is executed by the INSIGHT2+ program at runtime. This feature makes the knowledge base run faster and allows a larger knowledge base to run in a given amount of memory. However, the speed of compiling the knowledge base is proportional to the volume of the knowledge base. In other words, as the size of the knowledge base increases, the compile step is slow; this was experienced in this research.

The report functions are also useful feature of INSIGHT2+. These functions provide access to the line of reasoning when running a knowledge base. One of the important feature of INSIGHT2+ is the capability for data transfer between a knowledge base and a data base or
external programs. Within INSIGHT2+, the ACTIVATE, the SEND, and the RETURN commands are used to interface with external programs (PASCAL, C, or FORTRAN) or data base (dBASE II or III). All in all, it has been found that INSIGHT2+ is a convenient and easily used expert system shell in this research, although the compiler speed is slow.

INSIGHT2+ is written in Turbo PASCAL and runs on an IBM PC XT or AT under MS-DOS version 2.0 or later operating system. It requires a minimum of 448K to have access to the full functionality of the INSIGHT2+ system. Thus, the recommended configuration for using it is 640K and a hard disk drive.

Personal Consultant Plus [72] is a linear descendant of MYCIN [12], the great grandfather of so many of today's expert system shells. It provides a rule language for knowledge representation and a predominantly backward-chaining inference mechanism that allows the use of certainty factors. It also provides an extensive user interface both for the developer and end-user, which is the key to the whole system.

The ability to incorporate graphics into the consultation is another important benefit. Instead of confusing the user with a question such as
Is radioactive gas leaking from the crossbar manifold tubing bracket junction (A), or the extended transverse connection shaft (B)?

The system can present images of the two items of equipment.

The fundamental knowledge structure used is production rules. Although the reasoning strategy is primarily backward-chaining, there are ways of implementing forward-chaining.

Personal Consultant Plus is written in LISP and runs on the IBM AT and the TI Explorer symbolic processor with at least 512K and hard disk.

M.1 [24], a PROLOG version of Teknowledge's microcomputer-based expert system building tool, also implements control procedures and rule structures similar to those found in EMYCIN [73], which is a domain-independent version of MYCIN [12]. Consequently, M.1 has an advantage similar to that of Personal Consultant Plus, namely that EMYCIN applications can be transferred to the M.1 microcomputer environment. SACON [74] is a demonstration of an M.1 application, which advises engineers on the use of a complex structural analysis simulation program.

M.1 operates under MS-DOS 2.0 or later version. Also, the M.1 was converted to C for the improvement of its performance.
TIMM [75] combines a knowledge engineering tool capable of automatic rule definition with an inference mechanism that uses inexact reasoning. Definition of the problem domain can be carried out in terms of a set of attributes and their possible values or range of values. After the problem domain definition, an expert instructs TIMM systematically to form production rules. The system uses the problem domain definition to generate plausible event conditions for which the expert presents the advice or action.

The inference engine uses partial matching of rule conditions and a metric for determining similarity of conditions to determine which rules should be executed. The partial matching permits cases for which not all of the conditions of a rule are matched exactly by the data of the current situation. As a result, the TIMM inference mechanism is analogical; if the current situation is similar to conditions described in the rule base, the rules that match it best will be used to infer advice on the problem.

TIMM is written in FORTRAN 77 and runs on the IBM PC/XT or AT with the corresponding math coprocessor chip; 8087 or 80287 math chip. On the IBM PC, it requires 640K memory and hard disk. Since it is written in FORTRAN, it runs on every computer system with FORTRAN capabilities.
4.3 Disadvantages and Advantages of Implementing Expert Systems on Microcomputers

The important issue related to the implementation of expert systems on microcomputers is the size of the knowledge base which can be handled by a microcomputer-based expert system tool. Microcomputer-based expert systems face the memory and data storage restrictions imposed by the microcomputer hardware, and thus face limitations on the size of problems that can be addressed. The maximum memory that can be handled by the system is 640K and this memory limitation, in turn, limits the feasibility of microcomputer-based expert system applications. As a result, it is important for a knowledge engineer to estimate the size of the knowledge bases required for a problem domain. It is difficult to make such estimates, even though he has experience in expert system applications. However, the number of rules in the knowledge base depends on the problem domain of some technical field. Depending on the problem domain, we can, however, build low-cost, simple prototype expert systems on microcomputers, before venturing into a major developmental effort. The microcomputer-based expert system tools described above are adequate to handle on the order of several hundred rules. This is adequate not only to build low-cost prototype expert systems, but also to
develop complete expert systems for a number of applications. As a matter of fact, today's microcomputer-based expert systems offer real power and, more importantly, the ability to develop an expert system on one machine and deliver it for use on another. Furthermore, the appearance of the new operating system of OS/2 [71] and the trend of decreasing hardware costs will allow the microcomputer-based expert systems to keep pace with the development of larger expert system applications. The issue of the size of the knowledge bases will remain only as important as it is for applications developed on larger mainframe computers.

When the only device for data storage in microcomputers was 5.25 inch floppy disks, the microcomputer performance was disappointing because of small data storage capacity, although the machine did well at processing data. However, the problem was solved by the availability of a hard disk. Several advantages of a hard disk microcomputer system over a floppy disk system, including speed, capability, reliability, and an ability to support a great number of terminals, greatly enhanced the feasibility of a microcomputer-based expert system application.

Another issue relates to the computing power of microcomputer-based expert systems in LISP type architecture. The computational power of a microcomputer
depends on the number of bits the microprocessor processes at one time and the frequency of the processor's electronic clock (the number of steps a computer processes in a second). As the number of bits processed in a step increases, an operation can be completed in fewer machine cycles. As the frequency increases and as there are more cycles per second, the computer operations can be completed faster. It is found that symbolic processing on a 16 bit microprocessor is relatively slow. As a result, some microcomputer-based expert system tools, originally developed using LISP or PROLOG type architecture, have been ported to the C environment to increase the speed or computing power. However, now 32 bit microprocessor chips are available for the microcomputers and these are becoming the standard general-purpose microprocessors for AI in the microcomputers. This trend has solved the problem of microcomputer-based expert system in LISP or PROLOG type architecture.

The primary motivations for implementing an expert system on a microcomputer are the low cost, availability, and portability of the system. The low cost and easy availability of microcomputer systems make it possible to distribute inexpensive multiple copies of the same system. For example, this makes it possible for an electric company
to distribute multiple copies of an expert system for use at
different units without paying for expensive minicomputer
installations, which typically costs around $ 50,000 to
$100,000. A complete microcomputer costs less than $10,000,
even if the system is composed of 32 bit microprocessors.

With regard to portability, the small size and robust
design of microcomputer systems make them appropriate for
applications where physical space is limited and rough
environments may be encountered. For example, if we intend
to develop an expert system capable of monitoring and
evaluating the turbine-generator of a nuclear power plant
for the purpose of predictive maintenance, a portable
microcomputer system can be easily installed in the system
board. This makes interactive, on-line knowledge
engineering with expert systems at remote sites more
feasible.

In conclusion, the low cost, availability, the
portability, and enhanced capability of the microcomputer
system make it a very useful tool for prototyping large
expert systems and for developing many practical smaller
systems.
5. XSCD - AN EXPERT SYSTEM FOR XENON SPATIAL CONTROL

5.1 Reactor Control Model for Load-Follow of a PWR

The basic functional requirement of a nuclear power plant is to meet demand. This demand represents a load to the plant turbine-generator. To meet this load demand, the steam generator must respond with the correct flow of preconditioned steam to the turbine. In a PWR plant, the correct steam conditions and flow depend on the reactor output. Therefore, the PWR must have capabilities to respond to changes in turbine-generator load demand.

There are two aspects of controlling PWRs to follow the load demand: power level and power shape control. The power level control requires changes in reactivity. Three primary mechanisms to provide reactivity control in a PWR are full-length control rod (FLCR), chemical shim (soluble boron) and moderator temperature. The FLCRs, made of neutron absorbing materials, are assembled into clusters and manipulated as groups or banks of clusters. Each bank consists of four to eight control rods, which are symmetrically located around the center of the core. Due to safety considerations, FLCR banks have an insertion limit. The reactor operator keeps the FLCR bank above some minimum axial position to assure sufficient rod worth to shut the
reactor down properly from a given power level. In addition, the FLCR bank insertion limit can minimize the consequence of a rod ejection accident and help to maintain a more suitable axial power distribution.

The primary purpose of the chemical shim is to provide reactivity compensation for fuel burnup. Additionally it is also used to control changes in reactivity due to the effects on reactivity of xenon changes that arise during power level changes. Since the chemical shim is dissolved in the reactor coolant system (RCS) and evenly dispersed, a more even axial power distribution results than would exist if control rods were used alone. The rate of the boration or dilution process depends on the volume and concentration of coolant in the primary loop, volume and concentration of the volume control tank (VCT), charging pump capacity, and boric acid concentration. As a result, the chemical shim is a much slower process than control rods.

Moderator temperature control is a very useful control mechanism for short-term power level change. To produce suitable steam generation characteristics, PWRs operate at a programmed average coolant temperature that linearly varies with power level. Using both rod motion and chemical shim changes, the programmed coolant temperature can be maintained. However, near the end of cycle, boron dilution
may be difficult due to the low soluble boron concentration in the RCS. Consequently, it may not be possible to insert control rods to maintain the programmed coolant temperature and constant power level. Therefore, it is desirable at times, especially near the end of cycle, to allow coolant temperature to deviate from its programmed value and to utilize the reactivity feedback effect to produce a desired power level change.

The power shape control is also very significant. The average power density of the reactor is limited by the power peaking in the axial and radial core planes because safety limits on fuel performance affect the hottest point of the core. Axial power peaking is sensitive to the axial xenon distribution and control rod insertions which are influenced by reactor operator control action during power maneuvering situations. The PLCRs were originally designed to suppress axial xenon oscillations in PWRs. The PLCRs are thus used to control the axial power shaping.

There are other operational aspects involved in using PLCRs. When a PLCR is inserted to the bottom of the core and a PLCR is partially inserted, a condition called a "power-pinching" effect is created. This effect must be avoided because it could create an axial power peaking, which could cause fuel damage. Since the PLCRs can move
several times per day to control axial xenon oscillations under steady state and, even more often, during power maneuvering situations, local power density changes in the neighboring fuel rods can become a problem. In addition, if the PLCRs remain in a given region for a significant portion of the fuel cycle, rather large power peaking can result when the PLCRs are removed or moved to a different location.

To study the capability of an expert system for the control of axial xenon oscillations in PWRs, a control model of a typical PWR core is needed to simulate the axial xenon oscillations. The simulation core control model should be simple enough to be computationally feasible. A high-order core model does not necessarily mean that the model is better for control purposes than a lower-order core model as Frogner and Grossman [76] pointed out. Considering this idea, a PWR core control model described by a one-group diffusion model with moderator temperature and xenon-iodine feedbacks [37] was chosen. Furthermore, since coolant inlet temperature control is desirable for the reactivity control of the core, the coolant inlet temperature control was added to the PWR core coolant model.

The reactor core control model based on one-group, one-dimensional diffusion theory with moderator temperature and xenon-iodine feedbacks [37] is
\[
\frac{1}{V} \frac{\partial \Phi(r, \theta, z, t)}{\partial t} = \nabla \cdot \nabla \Phi - \Sigma_a \Phi + \nu \Sigma_f \Phi - \sigma_x X \Phi - \Sigma_c \Phi \\
- C_B \sigma_B B_B N_w \Phi - a_m \nu \Sigma_f (T_C - \bar{T}_C) \Phi
\]  
(1a)

\[
\frac{\partial I(r, \theta, z, t)}{\partial t} = \gamma_I \Sigma_f \Phi - \lambda_I I
\]  
(1b)

\[
\frac{\partial X(r, \theta, z, t)}{\partial t} = \gamma_X \Sigma_f \Phi + \lambda_I I - \lambda_X X - \sigma_X X \Phi
\]  
(1c)

where

- \( V \) = neutron velocity [cm/sec]
- \( \Phi \) = neutron flux [1/cm\(^2\)sec]
- \( t \) = time [sec]
- \( D \) = neutron diffusion coefficient [cm]
- \( \Sigma_a \) = macroscopic absorption cross section [1/cm]
- \( \Sigma_f \) = macroscopic fission cross section [1/cm]
- \( \nu \) = number of neutron per fission
- \( \sigma_x \) = microscopic neutron absorption cross section of Xe-135 [cm\(^2\)]
- \( X \) = Xe-135 concentration [1/cm\(^3\)]
- \( \Sigma_c \) = control rod absorption cross section [1/cm]
- \( C_B \) = conversion factor for boron concentration from ppm to boron-10 number density
- \( \sigma_B \) = microscopic neutron absorption cross section of boron-10 [cm\(^2\)]
- \( B_B \) = boron concentration [ppm]
- \( N_w \) = water molecule number density [1/cm\(^3\)]
• \( a_m \) = moderator temperature reactivity coefficient 
  \[
  [(\delta k/k)/°C]
  \]
• \( T_C \) = coolant temperature \(^{°C}\)
• \( \overline{T}_C \) = coolant reference temperature \(^{°C}\)
• \( I \) = I-135 concentration \([l/cm^3]\)
• \( \gamma_I \) = fission yield of I-135
• \( \lambda_I \) = decay constant of I-135 \([1/sec]\)
• \( \gamma_X \) = fission yield of Xe-135
• \( \lambda_X \) = decay constant of Xe-135 \([1/sec]\)

As shown by Cho and Grossman [37], moderator temperature feedback can be expressed as a change in the absorption cross section. The control rod absorption cross section \( \Sigma_C(z,t) \) consisting of a full-length control bank and a part-length control bank is given by

\[
\Sigma_C(r,\theta,z,t) = A_T \Sigma_{Cf} \left[ \sum_{i=1}^{n} \frac{1}{r} \delta(r - r_i) \delta(\theta - \theta_i) \right] + \sum_{j=1}^{m} \delta(\theta - \theta_j) \left[ H(z - z_2(t)) - H(z - z_2(t) - k) \right]
\]

where

• \( A_T \) = effective cross sectional area of a control rod cluster
• $\Sigma_{cf}$ = absorption cross section of full-length control rod
• $\Sigma_{cp}$ = absorption cross section of part-length control rod
• $r_i, \theta_i$ = polar coordinates of $i$th full-length control rod cluster
• $r_j, \theta_j$ = polar coordinates of $j$th part-length control rod cluster
• $z_1(t)$ = position of the tip of full-length control bank
• $z_2(t)$ = position of the tip of part-length control bank
• $k$ = length of absorbing material in the part-length control rod
• $n$ = number of full-length control rod cluster in a bank
• $m$ = number of part-length control rod cluster in a bank
• $\delta$ = Dirac delta function
• $H$ = Heaviside unit step function

Using the separation of variables technique in the radial and axial directions,

\[ \Phi(r,\theta,z,t) = J_0(B_r r)\psi(z,t) \] (3a)
\[ I(r,\theta,z,t) = J_0(B_r r)I(z,t) \] (3b)
Substituting Eq. (3a) into (1) and integrating over the \( r \) and \( \theta \) directions,

\[
\frac{1}{V} \frac{\partial \psi(z,t)}{\partial t} = D \frac{\partial^2 \psi}{\partial z^2} [\nu \Sigma_f - \Sigma_a - DB_r^2 + am \rho \Sigma_f T_C] \quad (4a)
\]

\[
\psi(z,t) = \sigma_x X(z,t) \psi(z,t) - \Sigma_c(z,t)
\]

\[
\psi(z,t) = C_b \sigma_b B_b(t) N_w(z,t) \psi(z,t)
\]

\[
- a_m \rho \Sigma_f T_C(z,t) \psi(z,t)
\]

\[
\frac{\partial I(z,t)}{\partial t} = \gamma \Sigma_f \psi(z,t) - \lambda I(z,t) \quad (4b)
\]

\[
\frac{\partial X(z,t)}{\partial t} = \gamma \Sigma_f \psi(z,t) + \lambda I(z,t) - \lambda X(z,t) - \bar{\sigma}_x X(z,t) \psi(z,t) \quad (4c)
\]

where

\[
\bar{\sigma}_x = \frac{\int_0^R r J_0^2(B_r r) dr}{\int_0^R r J_0(B_r r) dr} \quad \sigma_x \quad (4d)
\]

and
The coolant temperature distribution $T_C(z,t)$ can be defined by the internal energy balance relation as follows:

$$W_{ch} C_p dT_C(r,z,t) = A_{ch} \sum J_0(B_{rj} \psi(z,t)dz$$  \hspace{1cm} (5a)$$

where
- $W_{ch}$ = coolant mass flow rate per channel
- $C_p$ = heat capacity of coolant [J/kg °C]
- $A_{ch}$ = cross sectional area corresponding to $W_{ch}$
- $\varepsilon$ = recoverable energy per fission [J/fission]

Thus, for the all coolant channels, Eq. (5a) can be redefined as follows:

$$W_{ch} C_p dT_C(r,z,t) = 2\pi r dr \varepsilon \sum J_0(B_{rj} \psi(z,t)dz$$  \hspace{1cm} (5b)$$

Integrating Eq. (5b) over the $r$ direction,

$$T_C(z,t) = \frac{2\pi \varepsilon \sum}{W_{ch}} \int_0^R \int_0^z J_0(B_{rr}) dr \psi(z',t) dz' + T_{in}(t)$$  \hspace{1cm} (5c)$$
where

\[ T_{\text{in}}(t) = \text{coolant inlet temperature at time } t \ [\degree \text{C}] \]

Similarly, the coolant density distribution at steady state, \( N_w(z,0) \), can be defined by using the coolant thermal expansion coefficient \( \beta \) at constant pressure as follows:

\[ \beta = - \frac{1}{N_w(0,0)} \frac{\partial N_w(z,0)}{\partial T_c(z,0)} \]  \hspace{1cm} (6)

where

\[ T_c(z,0) = \text{coolant temperature distribution at steady state} \]

Based on Eq. (6), \( N_w(z,0) \) is

\[ N_w(z,0) = N_w(0,0) \exp\left[ \frac{2\pi \Sigma_f \beta}{W_C P} \int_0^R \int_0^Z \Psi(z')dz' \right] \]  \hspace{1cm} (7)

Approximating the exponential function of Eq. (7) and using Eq. (5c) at steady state, Eq. (7) can be rewritten as

\[ N_w(z,0) = N_w(0,0)[1 - \beta(T_c(z,0) - T_{\text{in}}(0))] \]  \hspace{1cm} (8)

where

\[ T_{\text{in}}(0) = \text{coolant inlet temperature at steady state} \]
It is clear that Eq. (4) is a system of coupled nonlinear partial differential equations. Since our interest is to make a simulation model which controls small perturbations around some steady state, the nonlinear system equations can be linearized as follows:

\[ \frac{\psi(z,t)}{t} = \psi(z,0) + \delta \psi(z,t) \]  
\[ I(z,t) = I(z,0) + \delta I(z,t) \]  
\[ X(z,t) = X(z,0) + \delta X(z,t) \]  

(9a)  
(9b)  
(9c)

Therefore, Eq. (9) requires two steps to find the simulation solutions: steady state solutions and perturbation solutions.

To derive the steady state solutions, we set the time derivative terms in Eq. (4) equal to zero, substitute \( T_C(z,0), N_w(z,0) \) and \( X(z,0) \) in Eq. (4a) using Eqs. (4c), (5c) and (7), and then rewrite Eq. (4a) in operator form by introducing dimensionless variables

\[ y = \frac{z}{L} \]

\[ u = \left( \frac{\sigma_x}{\lambda_x} \right) \psi(z,0) \]  

(10)

where

- \( L \) : reactor height [cm]

Thus, the steady state equation in operator form becomes
\[ \begin{align*}
\bar{R} u + \lambda u &= g(y,u), \quad 0 \leq y \leq 1 \\
\text{where} \\
\bar{R} &= \frac{d^2}{dy^2} - p(y) \\
p(y) &= \frac{L^2}{D} \left[ DB_f + \Sigma_c(y,0) + a_m \Sigma_f (T_{in}(0) - T_c) \right. \\
&\quad \left. + C_B \sigma_B B_B(0) N_w(y,0) \right] \\
\lambda &= \frac{L^2}{D} [\nu \Sigma_f - \Sigma_a] \\
g(y,u) &= Q \frac{u^2}{1 + u} + Ru \int_0^1 B_f(r) \text{d}r \quad u(y') \text{d}y' \\
Q &= \frac{L^2}{D} (\gamma_f + \gamma_x) \Sigma_f \\
R &= \frac{L^3}{D} \left( \frac{2 \pi \rho \Sigma_f}{\rho_c} \right) [a_m \nu \Sigma_f - C_B \sigma_B B_B(0) N_w(y,0) \beta]
\end{align*} \]

The boundary conditions are

\[ \begin{align*}
u(0) &= 0 \\
u(1) &= 0
\end{align*} \]

Because of the nonlinear term in the right hand side of Eq. (11a), Eq. (11a) can not be directly solved for the steady state solution. The general procedure to handle the nonlinear eigenvalue problems is to rewrite the differential term of Eq. (11b) in finite difference form, cast the resulting system of difference equations into matrix form, and then to obtain iterative solutions of the finite
difference equations on a computer. However, there are still two problems even if we use the iterative method. One relates to how the full- and part-length control rod absorption cross sections in the core are defined and the other is the selection of the fundamental eigenfunction or initial function for the iterative process.

The control rod absorption cross section in Eq. (4e) is composed of the Heaviside unit step functions which are piecewise continuous. Thus, according to the full- and part-length control rod positions in the core, $\Sigma_c(y,0)$, can be represented as

$$\Sigma_c(y,0) = \begin{cases} 
0, & 0 \leq y \leq y_1 \\
\Sigma_{cp}, & y_1 \leq y < y_2 \\
0, & y_2 \leq y < y_3 \\
\Sigma_{cf}, & y_3 \leq y \leq 1
\end{cases}$$

(13)

As proved in reference [35], bifurcation theory [52,53], which has been applied to several nuclear reactor problems [54-56], can be used to derive the positive fundamental eigenfunction $u_0(y)$ of the nonlinear eigenvalue problem of Eq. (11a). By the definition of Dean and Chambre [57], the bifurcation point of Eq. (11a) is the fundamental eigenvalue $\lambda$, say $\lambda_0$, of the linearized equation around a solution of Eq. (11a) for $\lambda_0$, say $u_0(y)$,

$$\mathcal{R}\psi(y,0) + \lambda_0\psi(y,0) = g_u(y,u_0(y))\psi(y,0)$$

(14)
Since \( u(y) = 0 \) for \( 0 \leq y \leq 1 \) satisfies Eq. (11a) for any \( \lambda \), \( u_0(y) = 0 \) can be a solution of Eq. (11a) for \( \lambda_0 \), even though the trivial solution has no physical meaning. Therefore, we see \( g_u(y,0) = 0 \) in Eq. (14) and the fundamental eigenvalue and eigenfunction, \( \lambda_0 \) and \( \psi_0(y,0) \), can be easily found from the following equation,

\[
\partial \psi(y,0) + \lambda \psi(y,0) = 0 \quad (15a)
\]

\[
\psi(0,0) = \psi(1,0) = 0 \quad (15b)
\]

Physically, the fundamental eigenfunction \( \psi_0(y,0) \) represents the power distribution in the core without the xenon and coolant temperature feedback effect, since Eq. (15a) does not contain the nonlinear term, which stand for the xenon and coolant temperature feedback effect in the core.

Therefore, the steady state solution \( \psi(y,0) \) can be obtained by the iterative scheme with Eqs. (13) and (15a). To accelerate the convergence of the iteration scheme of Eq. (11a), it is desirable to use the well known, successive overrelaxation (SOR) method with an acceleration parameter.

A computer program, STEADY.FOR, was written to obtain the steady state solution of Eq. (11a) on a microcomputer using the IBM professional FORTRAN compiler, based on the procedure described above. The SOR algorithm was introduced into the STEADY.FOR program to accelerate the iterative
scheme. Appendix A lists the source code of the STEADY.FOR program.

The next step to get the space- and time-dependent solutions of Eq. (9) is to derive the perturbation terms in Eq. (9). Linearizing Eq. (4) using

\[
\psi(z,t) = \psi(z,0) + \delta \psi(z,t) \quad (16a)
\]
\[
B_b(t) = B_b(0) + \delta B_b(t) \quad (16b)
\]
\[
N_w(z,t) = N_w(z,0) + \delta N_w(z,t) \quad (16c)
\]
\[
X(z,t) = X(z,0) + \delta X(z,t) \quad (16d)
\]
\[
\Sigma_c(z,t) = \Sigma_c(z,0) + \delta \Sigma_c(z,t) \quad (16e)
\]
\[
T_{in}(t) = T_{in}(0) + \delta T_{in}(t) \quad (16f)
\]
\[
T_c(z,t) = T_c(z,0) + \delta T_c(z,t) \quad (16g)
\]

and neglecting steady state terms, and second and higher order deviations, we obtain

\[
\frac{1}{V} \frac{\partial \delta \psi}{\partial t} = D \frac{\partial^2 \delta \psi}{\partial z^2} + (\nu \Sigma_f - \Sigma_a - DB_b^2 + a_m \Sigma_f T_c) \delta \psi + \frac{\partial \delta I}{\partial t} - \gamma I \Sigma_f \delta \psi - \lambda I \delta I + \lambda I \delta I - \nu \Sigma_f \delta \psi - \lambda I \delta I - \nu \Sigma_f \delta \psi - \lambda I \delta I - \nu \Sigma_f \delta \psi - \lambda I \delta I - \nu \Sigma_f \delta \psi - \lambda I \delta I - \nu \Sigma_f \delta \psi - \lambda I \delta I - \nu \Sigma_f \delta \psi - \lambda I \delta I
\]

where
\[ \delta \Sigma_c(z,t) = \frac{A_r}{2\pi \int_0^R r J_0(B_r r) dr} \left[ \sum_{i=1}^n \left( B_{r_i} \right) \delta(z - z_1(0)) \delta z_1(t) \right] \]
\[ + \sum_{j=1}^m B_{r_j} \left( \delta(z - z_2(0)) - \delta(z - z_2(0) - 1) \right) \delta z_2(t) \]
\[ \delta T_c(z,t) = \frac{2\pi \Sigma_f}{\omega c_p} \left( r J_0(B_r r) \right) \int_0^z \delta \psi(z',t) dz' + \delta T_{in}(t) \]
\[ \delta N_w(z,t) = -\beta N_w(z,0) \left( \frac{2\pi \Sigma_f}{\omega c_p} \right) \int_0^z r J_0(B_r r) dr \int_0^z \delta \psi(z',t) dz' + \delta T_{in}(t) \]

\[ \delta: \text{dirac delta function} \]

and the boundary and the initial conditions are

\[ \delta \psi(0,t) = \delta \psi(L,t) = 0 \] (18a)
\[ \delta \psi(z,0) = \delta I(z,0) = \delta X(z,0) = 0 \] (18b)

Considering the properties of the eigenfunctions, we are able to expand \( \delta \psi(z,t), \delta I(z,t), \) and \( \delta X(z,t) \) in eigenfunctions of the Helmholtz equation \( \psi_i(z) \):

\[ \delta \psi(z,t) = \sum_{i} a_i(t) \psi_i(z) \] (19a)
\[
\delta I(z,t) = \sum_i b_i(t)\psi_i(z) \\
\delta X(z,t) = \sum_i c_i(t)\psi_i(z)
\]

where \(\psi_i(z)\) are the eigenfunctions of

\[
\frac{d^2\psi}{dz^2} + 2k_i\psi_i(z) = 0, \quad \psi_i(0) = \psi_i(L) = 0
\]

Thus,

\[
\psi_i(z) = \sin \frac{i\pi z}{L} \quad i = 1, 2, 3 \ldots
\]

Using Eq. (19) with (20), the perturbation solutions of Eq. (9) resulting from perturbations in the core can be obtained, if \(a_i(t), b_i(t),\) and \(c_i(t)\) are known in Eq. (19). In order to find \(a_i(t), b_i(t),\) and \(c_i(t)\), substitute Eq. (19) into (17) and form the inner product with \(\psi_j\). Based on a two-term expansion, the following system of differential equations results:

\[
\begin{bmatrix}
a_1(t) \\
a_2(t)
\end{bmatrix}
= A
\begin{bmatrix}
a_1(t)
\end{bmatrix}
+ B
\begin{bmatrix}
b_1(t) \\
b_2(t)
\end{bmatrix}
+ C
\begin{bmatrix}
c_1(t) \\
c_2(t)
\end{bmatrix}
\begin{bmatrix}
\delta Z_1(t) \\
\delta Z_2(t) \\
\delta B_b(t) \\
\delta T_{in}(t)
\end{bmatrix}
\]
\[
\begin{bmatrix}
    b_1(t) \\
    b_2(t) \\
    c_1(t) \\
    c_2(t)
\end{bmatrix}
= D \begin{bmatrix}
    b_1(t) \\
    b_2(t) \\
    c_1(t) \\
    c_2(t)
\end{bmatrix}
+ E \begin{bmatrix}
    a_1(t) \\
    a_2(t)
\end{bmatrix}
\]

with initial conditions:
\[
\begin{bmatrix}
    a_1(0) \\
    a_2(0) \\
    b_1(0) \\
    b_2(0) \\
    c_1(0) \\
    c_2(0)
\end{bmatrix}
= \begin{bmatrix}
    0 \\
    0 \\
    0 \\
    0 \\
    0 \\
    0
\end{bmatrix}
\]

where
\[
A = \begin{bmatrix}
    A_{11} & A_{12} \\
    A_{21} & A_{22}
\end{bmatrix}
\]
\[
B = \begin{bmatrix}
    B_{11} & B_{12} & B_{13} & B_{14} \\
    B_{21} & B_{22} & B_{23} & B_{24}
\end{bmatrix}
\]
\[
C = \begin{bmatrix}
    C_{11} & C_{12} & C_{13} & C_{14} \\
    C_{21} & C_{22} & C_{23} & C_{24}
\end{bmatrix}
\]
\[
D = \begin{bmatrix}
    D_{11} & D_{12} & D_{13} & D_{14} \\
    D_{21} & D_{22} & D_{23} & D_{24} \\
    D_{31} & D_{32} & D_{33} & D_{34} \\
    D_{41} & D_{42} & D_{43} & D_{44}
\end{bmatrix}
\]
Hence we now have a set of six coupled ordinary differential equations in time that describe the time-dependent control effects on the perturbation solutions of Eq. (19) in the core. Therefore, using both Eqs. (20) and (21), the perturbation solutions of Eq. (19) can be obtained and then the linearized time and space dependent simulation control model of Eq. (9) can be obtained.

A computer program, PREDICT.FOR, was written to calculate the time-dependent flux using Eqs. (9), (19), (20), and (21) on a microcomputer. Appendix B illustrates the evaluation of the elements of matrices [A], [B], [C], [D], and [E] in Eq. (21). Appendix C lists the source code of the PREDICT.FOR program.

5.2 Knowledge Representation for Load-Follow of a PWR

5.2.1 Introduction

As described in section 5.1, a load-follow strategy requires both reactor power level and reactor power shape control. The power level control requires changes in
reactivity. The control mechanism for reactivity changes in the simulation core model of a PWR are FLCRs, coolant inlet temperature, and boron concentration. For a load change, a negative or positive reactivity insertion can be made by moving the FLCRs downwards or upwards and increasing or decreasing the coolant inlet temperature. Furthermore, a dilution or boration process should be carried out in parallel to compensate for the negative or positive reactivity insertion resulting from the xenon concentration increase or decrease during the load change. At the same time, power shape control is required, since the movement of FLCRs results in an imbalance between the spatial distribution of the power and xenon density, which causes unacceptable local power peaks. In other words, since the FLCRs force power to the unrodded regions of the core, the axial power distribution will be changed and the balance between the spatial distribution of the power and xenon density will be broken.

The most practical and most often used method to control the power shape in a PWR has been the application of some form of constant axial offset control (CAOC) [35]. The CAOC is based on the goal of maintaining an axial offset (AO) value within a target AO (typically the steady state full power value). That is, the CAOC procedure tries to
operate the core in such a manner that a desired balance of the power generation in the top and bottom halves of the core is maintained. Such an operation always prevents large top and bottom xenon oscillations. These oscillations cause skewness of the power distribution and are relatively slow to decay.

The PLCRs are used to control the AO value in the simulation core control model of a PWR. Whenever the AO deviates from the target band, the PLCRs are moved to bring the AO within the target band. The direction of the PLCR movement depends on the direction that the power shape is shifting. For example, if the power shape shifts downwards in the core, the PLCRs also moves downwards to dampen the shifted portion of the power distribution.

Implementation of an advisory expert system to aid reactor operators in a load-follow operation requires that the control knowledge mentioned above be represented in terms of computer data structure and that procedures which allow intelligent manipulation of the computer data structure to make inference, be developed.

There are several knowledge representation techniques [6,20] that are used in expert systems: semantic networks, production system, frames, and logical expressions. Choosing the appropriate knowledge representation schemes is
very important because the inference engine consisting of problem-solving strategies heavily depends on the knowledge representation as Newell and Simon [65] have shown. The production system, based on rules called productions in the form of IF-THEN condition-action pairs, was selected for representing the control knowledge of the load-follow domain for the following reasons:

1. The production system is a modular knowledge representation scheme. An individual rule in the knowledge base can be used to represent an independent piece of knowledge for a domain. That is, each rule can be used to match each pattern for the reactor power level and shape.

2. The modularity of the production system allows a knowledge engineer to change or update rules easily in the knowledge base without having to worry about direct effect on the other rules.

3. Since the production system has a uniform data structure, all information encoded with the production rules can be more easily understood and represented than would be possible in the relatively free form of semantic networks.

In order for the inference engine to select control actions by emulating the reasoning process of an expert for
the load-follow operation, some state variables which represent the state of reactor power level and shape must be used in the inference mechanisms. For example, the difference between the desired and the current power level and the difference between the desired and the current AO at some time can be state variables representing the required amount of negative or positive reactivity and the necessary degree of power shift, respectively. The combination of the two state variables represents a pattern for the power level and shape control which indicates the amount of negative or positive reactivity insertion and the degree of the PLCRs movement required in the core. Therefore, the set of all possible combinations of each state variable can be used to represent all possible core state patterns occurring during the load-follow operation.

The definition of the set of all possible core state patterns is very important, since the AI approach to decide control actions essentially consists of searching for a particular core state pattern from all possible core state patterns. The pattern-matching strategy is thus useful in identifying a pattern and then deciding a control action based on this pattern. The action can be specified by IF (features pattern) THEN (take action) types of production rules. However, since the control action for the pattern is
a unique deterministic solution among the set of all possible solutions, known as the search space in AI terminology, it is difficult to select an exact amount of control input over time required to drive the reactor power level and shape towards the desired patterns. In other words, it is not practical to select a unique and exact amount of control input for each pattern.

A forward-chaining inference mechanism can be applied to govern the overall behavior of the control variables adaptively by recognizing the current patterns, deciding the control actions, predicting future behavior, and monitoring its execution repeatedly to ensure success. That is, the change of each control variable over time can be adapted to drive the reactor power level and AO towards the desired values. It is, however, difficult for the power level and AO to reach the desired values exactly by adapting the control variables heuristically. A target band about the desired power trace is thus needed in the power level control of the core. The target band for the power shape control has been already considered, since the control of the power shape is performed by the CAOC procedure in which a target AO band is necessarily required.

The following section 5.2.2 discusses the detailed core state variables and how to represent the pattern of the core power level and shape using the core state variables.
5.2.2 Core state representation and pattern-matching

The information available to define the core state is the current power distribution, the current power level, the current axial offset value, the desired power level trace, and the desired axial offset value. The current power distribution, power level, and axial offset value are generated from execution of the simulation core control model with the adjusted control rod position, boron concentration, and coolant inlet temperature. The desired axial offset value can be the axial offset value at steady state full power. The desired power level trace is the scheduled load demand which is provided in the data base as shown at Figure 1. From this information, core state variables can be defined to describe the state of the power level and shape in the core.

It is necessary to define two groups of state variables: one is for the power level description and the other is for the power shape description. The following variables are used to represent the power level and shape state in the core:

1. Relative power error (POERR)

\[ POERR(t) = \left( \frac{DP(t) - CP(t)}{DP(t)} \right) \times 100 \]
where DP and CP represent the desired and the current power at time \( t \), respectively.

2. Time rate of change of power level (PORATE)

\[
\text{PORATE} = \frac{\text{dp}}{\text{dt}}
\]

where \( \text{dp} \) and \( \text{dt} \) represent the change in power level and time between time at \( t-1 \) and time at \( t \), respectively.

3. Magnitude of the relative power error (APOERR)

\[
\text{APOERR} = |\text{POERR}|
\]

4. Relative axial offset error (AOERR)

\[
\text{AOERR}(t) = \left( \frac{\text{DA}(t) - \text{CA}(t)}{\text{DA}(t)} \right) \times 100
\]

where \( \text{DA} \) and \( \text{CA} \) represent the desired and the current axial offset values at time \( t \), respectively.

5. Time rate of change of axial offset (AORATE)

\[
\text{AORATE} = \frac{\text{da}}{\text{dt}}
\]

where \( \text{da} \) and \( \text{dt} \) represent the change in axial offset and time between time at \( t-1 \) and time at \( t \), respectively.

6. Magnitude of the relative axial offset error (AAOERR)
AAOERR = \vert AOERR \vert

The first variable, POERR, represents the relative power difference between the desired and the current power level. A positive value of POERR means that the core demands positive reactivity. On the other hand, the core demands negative reactivity for a negative value of POERR. The second variable, PORATE, describes the power level trend and gives more detailed power control information. For example, if POERR is negative and PORATE is positive, the core state requires more negative reactivity than that having a negative POERR and negative PORATE. The core power in the former condition is still increasing due to the positive slope of PORATE. Thus, this variable can be used to provide a fine power level control.

The variable APOERR, absolute value of POERR, is a measure of how much reactivity the core needs, based on the direction of POERR and PORATE, to reach the desired power level. The magnitude of the negative or positive reactivity controlling the power level can be determined in proportion to the magnitude of APOERR. However, it is not practical for an expert control system to respond to each different value of APOERR with a different control action, specifically because the range of the APOERR value is so wide. For example, if the APOERR ranges between 0 % and 20
% and was differentiated into 1 % bands, the expert control system would be required to consider 80 different kinds of control actions only for power level control, since there are two cases of POERR (negative or positive POERR), two cases of PORATE (negative or positive PORATE), and 20 cases of APOERR value.

The variable AOERR describes the direction of power shape shifting: negative (-) AOERR means that the power shape is shifted up compared to the desired shape and positive (+) AOERR corresponds to the power shape shifted down compared to the desired AO value, which is negative in this control model. The variable AORATE describes the power shape trend and gives a more detailed AO control information, as PORATE does in the power level control. A negative AORATE corresponds to the upward shifting of the power shape over time and a positive AORATE corresponds to the downward shifting of power level over time.

The variable AAOERR, absolute value of AOERR, is a measure of how much the power shape shifts upwards or downwards. The AAOERR also ranges widely and the determination of each specific control action for each different value of AAOERR is difficult. If we consider the same ranges and bands as for the APOERR, the expert control system should also prepare 80 different kinds of control
actions for the power shape control, since there are two cases of AOERR (positive or negative), two cases of AORATE (positive or negative), and 20 cases of the AAOERR value. Therefore, the search space for the power level and the power shape control has 6400 different patterns because the set of all possible patterns is the combination of 80 different control actions for the power level with 80 different control actions for the power shape. It is not practical and efficient for a microcomputer-based expert system to search for a solution from such a large search space. Furthermore, there is another serious problem even if we could implement an expert system having 6400 different patterns; this is the selection of a specific control action for each control variable for 6400 different patterns. To avoid the problems mentioned above for the APOERR and the AAOERR state variable, fuzzy sets were made for the APOERR and AAOERR value by dividing the range into a control band and a target band, respectively. The control band was again subdivided into two linguistic terms (big and medium control) to describe the more detailed magnitude of either power or AO error. Table 2 shows the error bands chosen for each pattern of APOERR and AAOERR. Figure 2 shows the subdivisions of the magnitude of power and AO error.
TABLE 2. Error bands for each pattern of APOERR and AAOERR

<table>
<thead>
<tr>
<th>ERROR BAND</th>
<th>APOERR</th>
<th>AAOERR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target band</td>
<td>0.0 % ≤(X^a) &lt; 1.0 %</td>
<td>0.0 % ≤(Y^b) &lt; 0.5 %</td>
</tr>
<tr>
<td>Medium control band</td>
<td>1.0 % ≤(X) &lt; 2.5 %</td>
<td>0.5 % ≤(Y) &lt; 1.0 %</td>
</tr>
<tr>
<td>Big control band</td>
<td>2.5 % ≤(X)</td>
<td>1.0 % ≤(Y)</td>
</tr>
</tbody>
</table>

\(^a\)X represents value of APOERR.
\(^b\)Y represents value of AAOERR.

The error bands were heuristically divided and then the values were chosen to identify the pattern of APOERR and AAOERR. The target AO band was chosen by considering that the target AO band must be sufficiently narrow so that the benefits of lowered power peaking factors can be obtained, yet it also must be sufficiently broad to allow the operator to make power change easily. Since the power level control is more sensitive than the AO control, a bigger target power control band was chosen for the former compared to the latter.

Figure 3 shows all possible combinations of state variables for power level patterns. The negative (-) or positive (+) sign of the state variables represents a feature and a symptom for the given core power level state. In other words, it can be used to infer the direction of
FIGURE 2. Subdivision of magnitude of power and AO error 

FLCRs (downwards or upwards), the necessity of a dilution or boration process, and the direction of coolant inlet temperature change (increase or decrease). The absolute value of the relative power error APOERR can be used to quantify the control adjustments of the FLCR position, born concentration, and coolant inlet temperature, based on the sign of both POERR and PORATE. According to the pattern of the APOERR band, different control adjustments for the FLCR bank position (DELZ1), boron concentration (DELB), and
coolant inlet temperature (DELT) should be assigned to the core state. A power level pattern inference process resulting in control adjustments for a power level pattern, based on IF-THEN production rules, is

\[
\text{POERR : negative(-)} \\
\text{IF PORATE : negative(-)} \\
\text{APOERR : X} \\
\text{DELZ1 : -Z1(X)} \\
\text{THEN DELB : -B(X)} \\
\text{DELT : +T(X)}
\]

where

- X represents a control band (big, medium, target) of APOERR.
- -Z1(X) represents a Z1(X) cm downward adjustment of FLCR bank position corresponding to X.
- -B(X) represents a downward adjustment of boron concentration corresponding to X.
- +T(X) represents an upward adjustment of coolant inlet temperature corresponding to X.

Table 3 shows all possible power level patterns and the corresponding control actions.
Power Prediction

POERR(−)  POERR(+)  

PORATE(−)  PORATE(+)  PARATE(−)  PARATE(+)  

APOERR Control Band  APOERR Target Band  APOERR Control Band  APOERR Target Band  APOERR Control Band  APOERR Target Band  APOERR Control Band  APOERR Target Band  

Big  Medium  Big  Medium  Big  Medium  Big  Medium

FIGURE 3. Combination of state variables for power level patterns
TABLE 3. All possible power level patterns and the corresponding control actions

<table>
<thead>
<tr>
<th>PATTERN1</th>
<th>PATTERN2</th>
<th>PATTERN3</th>
<th>PATTERN4</th>
</tr>
</thead>
<tbody>
<tr>
<td>IF</td>
<td>IF</td>
<td>IF</td>
<td>IF</td>
</tr>
<tr>
<td>POERR: (-)</td>
<td>POERR: (-)</td>
<td>POERR: (+)</td>
<td>POERR: (+)</td>
</tr>
<tr>
<td>PORATE: (-)</td>
<td>PORATE: (+)</td>
<td>PORATE: (-)</td>
<td>PORATE: (+)</td>
</tr>
<tr>
<td>APOERR: X</td>
<td>APOERR: X</td>
<td>APOERR: X</td>
<td>APOERR: X</td>
</tr>
<tr>
<td>THEN</td>
<td>THEN</td>
<td>THEN</td>
<td>THEN</td>
</tr>
<tr>
<td>DELZ1: -Z1(X)</td>
<td>DELZ1: -Z1(X)</td>
<td>DELZ1: +Z1(X)</td>
<td>DELZ1: +Z1(X)</td>
</tr>
<tr>
<td>DELB: -B(X)</td>
<td>DELB: -B(X)</td>
<td>DELB: +B(X)</td>
<td>DELB: +B(X)</td>
</tr>
<tr>
<td>DELT: +T(X)</td>
<td>DELT: +T(X)</td>
<td>DELT: -T(X)</td>
<td>DELT: -T(X)</td>
</tr>
</tbody>
</table>

All possible combinations of state variables for power shape patterns are shown in Figure 4. The negative (-) or positive (+) sign of the state variables also represents a feature and a symptom for the power shape state. A negative AOERR and a negative AORATE indicate that the power shape is shifted upwards and the power shape is also being shifted upwards over time, respectively. It thus infers the required direction of the PLCRs motion (upwards or downwards) to dampen the shifted power shape. The absolute value of the relative AO error AAOERR quantifies the control adjustment of the PLCR bank position, based on the feature and symptom of the power shape. The required control adjustment for the PLCR bank position (DELZ2) depends on the
position of the AAOERR band. One power shape pattern
inference process producing the adjustment of the PLCR bank
position, based on IF-THEN production rules, is

\[
\begin{align*}
\text{AOERR} & : \text{negative}(-) \\
\text{IF} & \quad \text{AORATE} : \text{negative}(-) \\
\text{AAOERR} & : Y \\
\text{THEN} & \quad \text{DELZ2} : +Z2(Y)
\end{align*}
\]

where

- \( Y \) represents a control band (big, medium, target)
of AAOERR.
- \(+Z2(Y)\) represents a \( Z2(Y) \) cm upward adjustment of
  PLCR bank position corresponding to \( Y \).

Table 4 shows all possible power shape patterns and the
 corresponding control actions.

So far, we have discussed the power level and the power
shape patterns, respectively. However, the load-follow
operation requires both a power level and a power shape
control simultaneously, which satisfy the technical
specification for the load-follow operation. Therefore, it
is necessary to simultaneously consider both the power level
FIGURE 4. Combination of state variables for power shape patterns
TABLE 4. All possible power shape patterns and the corresponding control actions

<table>
<thead>
<tr>
<th>PATTERN1</th>
<th>PATTERN2</th>
<th>PATTERN3</th>
<th>PATTERN4</th>
</tr>
</thead>
<tbody>
<tr>
<td>AOERR: (-)</td>
<td>AOERR: (-)</td>
<td>AOERR: (+)</td>
<td>AOERR: (+)</td>
</tr>
<tr>
<td>AORATE: (-)</td>
<td>AORATE: (+)</td>
<td>AORATE: (-)</td>
<td>AORATE: (+)</td>
</tr>
<tr>
<td>AAOERR: Y</td>
<td>AAOERR: Y</td>
<td>AAOERR: Y</td>
<td>AAOERR: Y</td>
</tr>
<tr>
<td>THEN</td>
<td>DELZ2: -Z2(Y)</td>
<td>DELZ2: -Z2(Y)</td>
<td>DELZ2: +Z2(Y)</td>
</tr>
</tbody>
</table>

and power shape pattern inferences which allow the operator to keep the power level and the AO error within the target band.

All possible combinations of state variables for power level and power shape patterns can be derived by combining all possible state variables for power level patterns with all possible state variables for power shape patterns. Using the state variables shown in Figure 3 and Figure 4, all possible patterns for the power level and power shape can be represented in the knowledge base using IF-THEN production rules. Since there are 12 cases of power level and power shape patterns respectively, there are 144 cases for all possible patterns for the power level and shape. Among these patterns, 16 cases exist in the target band.
because there are 4 cases each of power level and power shape patterns existing in the target band. Therefore, the total number of patterns would be 128 (144 - 16) cases. That is, the knowledge base is composed of 128 production rules. An example rule which infers control adjustments for a power level and shape is:

\[
\begin{align*}
\text{POERR} & : \text{negative}(-) \\
\text{PORATE} & : \text{negative}(-) \\
\text{IF} & \ APOERR : X \\
\text{AOERR} & : \text{negative}(-) \\
\text{AORATE} & : \text{negative}(-) \\
\text{AAOERR} & : Y \\
\text{DELZ1} & : -Z1(X) \\
\text{THEN DELB} & : -B(X) \\
\text{DELT} & : +T(X) \\
\text{DELZ2} & : +Z2(Y)
\end{align*}
\]
Heuristically, this rule translates as:

- **IF** the current power level is higher than the desired, the power level is increasing over time, the power difference resides in the X control band, the power shape is shifted upwards, the power shape is shifting upwards over time, and the degree of the power shape shifting resides in the Y control band,

- **THEN** move the FLCRs Z1(X) cm distance downwards, decrease the boron concentration an amount B(X), increase the coolant inlet temperature an amount T(X), move the PLCRs a distance Z2(Y) cm upwards.

As seen in this example rule, the pattern-matching process using the core state variables, is useful to infer the control direction, but is inefficient to infer an amount of control adjustment. In other words, this process clearly directs the FLCR and the PLCR motion, boron concentration change, and coolant inlet temperature change, yet has difficulty in quantifying each control adjustment for each control variable because this pattern-matching process produces a specific control action for a pattern of power level and shape. If we use only the pattern-matching process in load-follow operation, we must prepare 128 cases.
of specific control actions because the search space is composed of 128 cases of patterns of power level and shape. However, it is not practical to quantify each case of control action separately. Therefore, another inference mechanism, which infers the amount of control action for each control variable, is needed.
6. RULE LEARNING AND TESTING OF XSCD EXPERT SYSTEM

6.1 Introduction

Since the performance of the XSCD expert system depends to a large extent on the adequacy of the control inputs for control actions employed in the rules, determination of the control inputs is a very important process. In particular, determining control inputs that will cover a wide range of load-follow patterns is the most important issue of the heuristic approach.

There are two main factors that contribute to determine control inputs. One is the need to quantify linguistic terms like big, medium, and target error band to describe the magnitude of the power error (POERR) and the AO error (AOERR). The other is to take account of the relative ease of implementation of a control mechanism.

Since the control process uses four different control variables (DELZ1, DELB, DELT, and DELZ2), and each has its own character for power level and power shape control, it is necessary to weight these control variables. The control variables are heuristically weighted according to the error band and the character of each control variable.

In order to determine the control inputs, unit control input values for the control variables should be found in
advance. These unit values are the sizes of the "quanta" of control action that are to be applied. The control inputs to be coded into the rules are obtained by multiplying the unit control input values by the weight values assigned to the control variables. This is explained more clearly in section 6.3. Thus, a rule for a control action is the specification that a control system be adjusted by an amount equal to weight times unit. Determination of adequate unit values of DELZ1, DELB, DELT, and DELZ2 is an important rule learning process.

A trial and error strategy using two performance indexes (PIs) described in section 6.3 and a load-follow pattern was used to obtain the best unit control input values.

The selection of the weight values of control variables is described in section 6.2. The determination of unit control input values and the control inputs that are coded into the rules of the knowledge base are described in section 6.3. The XSCD expert system using these rules was tested for different types of load-follow pattern to demonstrate its capability. The results of these tests are presented in section 6.4.
6.2 Selection of Weight Values for Control Variables

There are three control variables for the power level control; DELZ1, DELB, and DELT. The amount of control action of each control variable depends on the pattern of the APOERR control band as shown in Table 3. For example, when the pattern of APOERR is in the big or medium control band, the amount of control action of each control variable should be big or medium. In other words, the amount of control action of each control variable should be weighted according to the pattern of the APOERR. In addition, an appropriate weighting of the control variables is necessary for each pattern of APOERR, since each control variable has different control characteristics in changing the power level. The change in power level can be initiated easily with movement of FLCRs. Thus, DELZ1 is the most convenient control variable since it is easily implemented by rapid movement of FLCRs.

The DELT is also a desirable control variable in changing power level, since the reactivity feedback effect of the coolant inlet temperature can be easily utilized to produce the desired power change. When coolant inlet temperature is varied by a certain amount, a change of the same magnitude is observed in the coolant average temperature. The change in the coolant average temperature
affects the reactivity of the core if the coolant temperature coefficient is not zero.

The net effect of DELB on the reactivity of the core is similar to that of DELT in the core. However, the change of DELB has several constraints such as the available dilution or boration rate and the undesirable generation of borated waste water. The rate of dilution or boration depends on several operational and system conditions such as the volume and concentration of the primary loop, the volume and concentration of the volume control tank (VCT), and charging pump capacity. In particular, generation of a large amount of waste water is not desirable economically. Therefore, a small weight on the DELB would result in a more desirable operation pattern.

The core state variable PORATE, representing the direction of power level change, gives more detailed information for weighting the amount of control action of each control variable. The only difference between pattern1 and pattern2 in Table 3 is the sign (or direction) of PORATE. Both patterns require addition of negative reactivity. This is because the current power level is higher than desired. However, pattern2 needs more negative reactivity than pattern1, since the positive sign of PORATE in pattern2 indicates that the power is increasing with
respect to time and will result in an increase of APOERR with time. Thus, pattern2 needs more weight on the amount of control action of each control variable than pattern1. A similar situation occurs between pattern3 and pattern4 in Table 3.

For the power shape control, only the DELZ2 control variable is used in the core control model. Thus, the DELZ2 can be weighted according to the pattern of the AAOERR. A big or medium weight on the DELZ2 is needed for the pattern of a big or medium control band of the AAOERR. Core state variable AORATE, representing the direction of power shape shift with time, also gives more detailed information in weighting DELZ2. As shown in Table 4, the difference between pattern1 and pattern2 is the sign (or direction) of AORATE. Since both patterns have a negative value of AOERR, both patterns need an upward movement of PLCRs. However, pattern1 needs more movement of PLCRs than pattern2 because the power shape of pattern1 is moving upwards with time. Thus, pattern1 needs more weight on the DELZ2 than pattern2. In the same manner, pattern4 needs more weight on the DELZ2 than pattern3.

Based on the ideas described above, weight values were assigned subjectively to define the meaning of an error band and the ease of implementation of a control mechanism. The assigned values for control variables are given in Table 5.
TABLE 5. The weight values assigned to control variables

<table>
<thead>
<tr>
<th>patterns</th>
<th>error band</th>
<th>DELZ1</th>
<th>DELB</th>
<th>DELT</th>
<th>DELZ2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATTERN1</td>
<td>big</td>
<td>12</td>
<td>4</td>
<td>10</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>target</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>PATTERN2</td>
<td>big</td>
<td>16</td>
<td>6</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>target</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>PATTERN3</td>
<td>big</td>
<td>16</td>
<td>6</td>
<td>12</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>target</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>PATTERN4</td>
<td>big</td>
<td>12</td>
<td>4</td>
<td>10</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>4</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>target</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

As seen in Table 5, the assigned values for pattern 4 are the same as for pattern 1, and the same for pattern 3 as for pattern 2. However, the values of DELZ1, DELB, and DELT are higher for pattern 2 and pattern 3 than those in pattern 1 and pattern 4. This is because POERR and PORATE have the same sign for pattern 2 and pattern 3, but different signs for
pattern1 and pattern4, as shown in Table 3. On the contrary, the value of DELZ2 is lower for pattern2 and pattern3 than for pattern1 and pattern4. The reason for this is that AOERR and AORATE have the same sign for pattern1 and pattern4, but different signs for pattern2 and pattern3, as shown in Table 4. The value of DELZ2 for the medium error band does not change since the AO control is less sensitive than the power level control.

It is possible to select another set of weight values. As mentioned above, the weight values were assigned subjectively, based on the error bands and the control character of a control mechanism. Control inputs to be coded into the rules are obtained by multiplying the weight values and unit control input values of the control variables. Using the set of weight values selected, a set of unit control input values can be obtained.

Section 6.3 describes a method to determine the unit control input values of DELZ1, DELB, DELT, and DELZ2, based on the weight values presented in Table 5.

6.3 Determination of Unit Control Input Values

A set of unit control input values can be obtained by repeatedly running the XSCD expert system. Thus, it is important to see how the XSCD expert system generates the
best control strategy before developing a set of unit control input values.

Figure 5 shows how a set of "best" control actions can be developed by the XSCD expert system. For example, if APOERR is not less than 1.0% and AAOERR is not less than 0.5%, as shown in Figure 5, the control variables are adjusted by control inputs from a rule corresponding to the existing pattern of power level and shape in the knowledge base. The core control code predicts new core state variables using the adjusted control variables and sends the new core state variables to the knowledge base. Then, the new APOERR and AAOERR are examined to see if they both are within the respective target band. If either APOERR or AAOERR is not within its target band, the new control variables are adjusted by control inputs from a rule corresponding to the new pattern. However, if both APOERR and AAOERR are within the respective target band, the control variables are saved as the best control action at a given time i on a file. This process is continued for the next time step.

This system adjusts the control variables at a given time step until both APOERR and AAOERR converge to the respective target band. In other words, using a forward-chaining inference process, the control variables were
Desired Power \( \text{i} \)

Core Control Code

Core State Variable

FORTRAN SUBSYSTEM

PRODUCTION RULE SYSTEM

Desired Power equal to 0.0

Saving Control Variable at \( \text{i} \)
then \( \text{i} = \text{i} + 1 \)

APOERR<1.0%
AACERR<0.5%

Rule Base Adjust Control Variable

END

FIGURE 5. Adaptation of control variables using forward-chaining inference
adapted to adjust APOERR and AAOERR to less than 1.0 % and 0.5 %, respectively. This is the feature of this system. However, this feature depends on the adequacy of control inputs (or unit control input values). If the control inputs are not adequate, APOERR and AAOERR will not converge rapidly to their respective target bands, and may even oscillate at some points in big or medium control bands.

In addition, this feature makes it possible for the XSCD expert system to cover a wide range of load-follow patterns. For example, suppose the rate of change of power level is smaller for load pattern X than for load pattern Y. Using the weight values presented in Table 5, one set of unit control input values could be obtained for pattern X and another set for pattern Y. When the unit control input values obtained from pattern X are used in pattern Y, the errors of power level and shape decrease steadily. This is because the unit control input values adjust the control variables a little bit at a time and thus direct APOERR and AAOERR toward their respective target band. However, when the unit control input values obtained from pattern Y are used in pattern X, the unit control input values might not cause APOERR and AAOERR to converge towards their respective target bands. This is because the adjustment in the control variables is too large, and the projected APOERR and AAOERR
can skip past their respective target bands. In other words, APOERR and AAOERR could oscillate continuously. Therefore, it is important to use a load-follow pattern that has a small rate of change of power level in developing a set of unit control input values.

To obtain a set of unit control input values of control variables which cover a wide range of load-follow patterns, the pattern E of Table 6 was chosen. It consists of lowering the power level from 100% to 80% at a rate of 2% per hour, leveling at 80% for six hours, and finally raising the power level from 80% to 100% at a rate of 2% per hour. The value of 2% used for the rate of change of power for pattern E was felt to represent a practical lower limit for this parameter.

Two performance indexes (PIs) have been used to determine a set of unit control input values. One relates to the square of the relative deviation of power level from the desired value over the total simulation time. This index can be used to search for a set of unit control input values which generates more accurate control strategies. It is defined in Eq. (22)

\[
PI(T) = \frac{\int_{0}^{T} [P_{t}(t) - P_{c}(t)]^2 dt}{\int_{0}^{T} [P_{c}(t)]^2 dt} \quad (22)
\]
TABLE 6. Examples of desired load-follow patterns

<table>
<thead>
<tr>
<th>Time (hr)</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E&lt;sup&gt;a&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>98</td>
</tr>
<tr>
<td>2</td>
<td>100</td>
<td>96</td>
<td>98</td>
<td>98</td>
<td>96</td>
</tr>
<tr>
<td>3</td>
<td>100</td>
<td>92</td>
<td>96</td>
<td>96</td>
<td>94</td>
</tr>
<tr>
<td>4</td>
<td>96</td>
<td>88</td>
<td>94</td>
<td>94</td>
<td>92</td>
</tr>
<tr>
<td>5</td>
<td>92</td>
<td>84</td>
<td>92</td>
<td>92</td>
<td>90</td>
</tr>
<tr>
<td>6</td>
<td>88</td>
<td>80</td>
<td>90</td>
<td>90</td>
<td>88</td>
</tr>
<tr>
<td>7</td>
<td>84</td>
<td>80</td>
<td>90</td>
<td>86</td>
<td>84</td>
</tr>
<tr>
<td>8</td>
<td>80</td>
<td>80</td>
<td>90</td>
<td>84</td>
<td>82</td>
</tr>
<tr>
<td>9</td>
<td>80</td>
<td>84</td>
<td>90</td>
<td>84</td>
<td>80</td>
</tr>
<tr>
<td>10</td>
<td>80</td>
<td>88</td>
<td>90</td>
<td>84</td>
<td>80</td>
</tr>
<tr>
<td>11</td>
<td>80</td>
<td>90</td>
<td>90</td>
<td>86</td>
<td>80</td>
</tr>
<tr>
<td>12</td>
<td>80</td>
<td>90</td>
<td>90</td>
<td>88</td>
<td>80</td>
</tr>
<tr>
<td>13</td>
<td>80</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>80</td>
</tr>
<tr>
<td>14</td>
<td>80</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>80</td>
</tr>
<tr>
<td>15</td>
<td>80</td>
<td>90</td>
<td>90</td>
<td>90</td>
<td>80</td>
</tr>
<tr>
<td>16</td>
<td>84</td>
<td>92</td>
<td>90</td>
<td>90</td>
<td>80</td>
</tr>
<tr>
<td>17</td>
<td>84</td>
<td>94</td>
<td>92</td>
<td>88</td>
<td>82</td>
</tr>
<tr>
<td>18</td>
<td>92</td>
<td>96</td>
<td>94</td>
<td>86</td>
<td>84</td>
</tr>
<tr>
<td>19</td>
<td>96</td>
<td>98</td>
<td>96</td>
<td>84</td>
<td>86</td>
</tr>
<tr>
<td>20</td>
<td>100</td>
<td>100</td>
<td>98</td>
<td>84</td>
<td>88</td>
</tr>
<tr>
<td>21</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>84</td>
<td>90</td>
</tr>
<tr>
<td>22</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>86</td>
<td>92</td>
</tr>
<tr>
<td>23</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>86</td>
</tr>
<tr>
<td>24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>86</td>
</tr>
<tr>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>86</td>
</tr>
<tr>
<td>26</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>86</td>
</tr>
</tbody>
</table>

<sup>a</sup>The unit control input values displayed in Table 7 were obtained from simulation of this load-follow pattern.

<sup>b</sup>Values represent power level (%) of full power.
where $P_t$ and $P_c$ represent the desired and current power level at time $t$, respectively.

The other performance index is the number of evaluations performed over the total simulation time. As mentioned above, adjustment of the control variables is repeated until both APOERR and AAOERR are within the respective target band at a given time step. Each set of adjusted control variables predicts a new APOERR and AAOERR. The number of evaluations performed over the total simulation time is equal to the number of adjustments of the control variables over the total simulation time. If a large number of evaluations are performed with a set of unit control input values, the unit control values need to be changed. This index can be automatically generated by a variable defined internally in the rules.

Using the load-follow pattern of type E presented in Table 6 and the weight values shown in Table 5, a set of unit control input values for the control variables were selected using the PIs. The process is described as follows:

1. A number of pairs of PI values were obtained by varying the unit control input value of DELT from 0.1 °C to 0.18 °C, with the unit control input values of DELZ1, DELZ2, and DELB fixed at 0.5 cm.
0.5 cm, and 0.05 ppm respectively. Figure 6(a) shows the results and illustrates that the values of DELT affect the performance index defined by Eq. (22). However, as shown in Figure 6(b), the number of evaluations performed does not change much with DELT. Using Figure 6(a), a DELT of 0.18 °C was chosen as a unit control input value. As seen in Figure 6(a), this is the highest value of DELT used and results in the minimum value for PI. In fact, this is the maximum value of DELT for which APOERR and AAOERR converge to their respective target bands when other unit control input values are fixed. If a unit value of DELT higher than 0.18 °C is used, either APOERR or AAOERR will not converge to the target band, but will oscillate.

2. With the unit control input values of DELZ1, DELZ2, and DELT fixed as 0.5 cm, 0.5 cm, and 0.18 °C respectively, this last number being the result from step 1, pairs of PI values were obtained by varying the unit value of DELB from 0.01 ppm to 0.08 ppm. The results are shown in Figure 7. The minimum PI in Figure 7(a) is about 1.6 times lower than that in Figure 6(a).
(a) Performance index based on Eq. (22)

(b) Number of evaluations performed

FIGURE 6. Performance indexes versus DELT variation
(a) Performance index based on Eq. (22)

(b) Number of evaluations performed

FIGURE 7. Performance indexes versus DELB variation
However, the number of evaluations performed is almost constant. That is, this performance index is not sensitive to the variation of the unit value of DELB. Therefore, from Figure 7(a), a unit control input value of DELB was chosen as 0.08 ppm at which the value of the square of relative deviation of power from the desired is lowest. Again, this value is also the maximum value of DELB for which APOERR and AAOERR converge to their respective target bands without oscillating around them.

3. DELZ1 and DELZ2 were not varied. The control rod drive mechanism is normally designed to lift the control rods in constant increments or steps by using a lift coil. Thus, the size of the control quantum is fixed by design for these variables.

The best unit control input values of the control variables obtained from this learning process are presented in Table 7. The best control inputs of the control variables are obtained by multiplying the best unit control input values given in Table 7 and the weight values given in Table 5. An example rule showing the best unit control input values and the weight values is shown in Figure 8. Flcr, Plcr, Bc, and Cit are the best unit control input
TABLE 7. The unit control input values of control variables

<table>
<thead>
<tr>
<th>patterns</th>
<th>DELZ1</th>
<th>DELB</th>
<th>DELT</th>
<th>DELZ2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATTERN1</td>
<td>-0.5</td>
<td>-0.08</td>
<td>0.18</td>
<td>0.5</td>
</tr>
<tr>
<td>PATTERN2</td>
<td>-0.5</td>
<td>-0.08</td>
<td>0.18</td>
<td>0.5</td>
</tr>
<tr>
<td>PATTERN3</td>
<td>0.5</td>
<td>0.08</td>
<td>-0.18</td>
<td>-0.5</td>
</tr>
<tr>
<td>PATTERN4</td>
<td>0.5</td>
<td>0.08</td>
<td>-0.18</td>
<td>-0.5</td>
</tr>
</tbody>
</table>

values and 12, 1, 4, and 10 are the weight values. Table 8 presents the best control inputs for the control variables.

Figure 9 shows the predicted power and AO for load pattern E used in this rule learning process. These results were obtained using the best control inputs presented in Table 8. Figure 10 represents the best control strategy required to obtain the results shown in Figure 9.
Rule: For power and axial offset pattern learning

IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod + Plcr * 1
AND Boron concentration := Boron concentration - Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

FIGURE 8. An example rule in the XSCD expert system

As mentioned earlier, the selection of a set of weight values is subjective. Thus, a set of unit control input values different from those in Table 7 could be obtained by using a set of weight values different from those given in Table 5 and repeating the process described above.

If the unit control input values are not changed and lower weight values than those given in Table 5 are assigned to the big or medium error bands, the APOERR and AAOERR
TABLE 8. The best control inputs of control variables

<table>
<thead>
<tr>
<th>patterns</th>
<th>error band</th>
<th>DELZ1</th>
<th>DELB</th>
<th>DELT</th>
<th>DELZ2</th>
</tr>
</thead>
<tbody>
<tr>
<td>PATTERN1</td>
<td>big</td>
<td>-6.0</td>
<td>-0.32</td>
<td>1.8</td>
<td>8.0</td>
</tr>
<tr>
<td></td>
<td>medium</td>
<td>-2.0</td>
<td>-0.16</td>
<td>0.36</td>
<td>2.0</td>
</tr>
<tr>
<td></td>
<td>target</td>
<td>-0.5</td>
<td>-0.08</td>
<td>0.18</td>
<td>0.5</td>
</tr>
</tbody>
</table>

|          | big        | -8.0  | -0.48 | 2.16 | 6.0   |
|          | medium     | -2.5  | -0.24 | 0.54 | 2.0   |
|          | target     | -0.5  | -0.08 | 0.18 | 0.5   |

|          | big        | 8.0   | 0.48  | -2.16| -6.0  |
|          | medium     | 2.5   | 0.24  | -0.54| -2.0  |
|          | target     | 0.5   | 0.08  | -0.18| -0.5  |

|          | big        | 6.0   | 0.32  | -1.8 | -8.0  |
|          | medium     | 2.0   | 0.16  | -0.36| -2.0  |
|          | target     | 0.5   | 0.08  | -0.18| -0.5  |

converge at a slower rate. This is because the unit control input values based on the weight values given in Table 5 give maximum rate of convergence of APOERR and AAOERR. On the other hand, if the assigned value is higher than those in Table 5, at least one of APOERR and AAOERR would oscillate.
FIGURE 9. Reactor power and AO value as a function of time for load-follow pattern E
FIGURE 10. The best control strategy corresponding to Figure 9
FIGURE 10. (Continued)
6.4 Testing of the XSCD Expert System

The unit control input values in Table 7 are determined using the core control model described in section 5.1, the specific values of the control and target error bands given in Table 2, and load pattern E of Table 6. Therefore, if we change one or more of them, a set of unit control input values different from those in Table 7 might be obtained.

To test the capability of this XSCD, each set of control inputs from Table 7 was coded and fixed into a rule corresponding to the pattern. Different types of load patterns presented in Table 6 were tested. Each case consists of lowering, leveling, or raising the power level at a rate of 2% or more per hour.

The results of the test performed on pattern A are presented in Figures 11 through 19. The system state at time $t = 0$ represents the initial 100% steady state power level and shape, initial or desired AO value, initial steady state xenon level and shape, and no adjustment of control mechanism. Figure 11(a) represents an initial control action taken by the knowledge base. Figure 11(b) represents the current power, AO, and control adjustment values at time $t = 0$. The current power and AO mean that the predicted power and AO values cause both APOERR and AAOERR to be
Control parameters decided by knowledge base

- Time step = 1.0
- Desired power step = 1.0
- Previous power = 1.0
- Previous axial offset = 1.0
- Full length control rod = 0.0
- Part length control rod = 0.0
- Boron concentration = 0.0
- Coolant inlet temperature = 0.0

(a)

Satisfied power, axial offset and control values

- Time = 1.0
- Desired power = 100.0
- Current power = 100.0
- Current xenon = 100.0
- Desired AO = -17.37
- Current AO = -17.37
- Full length control rod = 0.0
- Part length control rod = 0.0
- Boron concentration = 0.0
- Coolant inlet temperature = 0.0

(b)

FIGURE 11. Initial system state representation
FIGURE 11. (Continued)
within their respective target bands. The values for the control variables describe deviations from the respective steady state values of full and part length control rod bank positions, boron concentration, and coolant inlet temperature. Thus, zero values for all control variables mean that no control action is taken. Figure 11(c) shows the flux and xenon density distributions respectively at time $t = 0$, which correspond to the core state values of Figure 11(b). These are displayed on the computer screen following Figure 11(b).

Figures 12 through 16 show selected results of power and xenon distribution for pattern A during the implementation of XSCD. These results illustrate that there is no power oscillation and no power peaking during the load-follow maneuver. This latter point, the crucial one for successful load-follow, is illustrated by the fact that throughout the maneuver the power is below its initial value at each point in space.

Figure 17 shows that the predicted power and AO for pattern A closely follows the scheduled load demand within the target error band and maintains the desired power shape (essentially constant AO). Figure 17(b) shows that AO initially increases positively as power level decreases, then decreases below the desired value and finally returns.
FIGURE 12. Flux in units of $1.42 \times 10^{14}$/cm$^2$sec and xenon in units of $2.03 \times 10^{15}$/cm$^2$sec versus core height at time equal 4 hours and power equal 96% of full power
FIGURE 13. Flux in units of $1.42 \times 10^{14}$/cm$^2$sec and xenon in units of $2.03 \times 10^{15}$/cm$^3$sec versus core height at time equal 9 hours and power equal 80 % of full power.
FIGURE 14. Flux in units of $1.42 \times 10^{14}$/cm$^2$sec and xenon in units of $2.03 \times 10^{15}$/cm$^2$sec versus core height at time equal 17 hours and power equal 88 % of full power.
FIGURE 15. Flux in units of $1.42 \times 10^{14}/\text{cm}^2\text{sec}$ and xenon in units of $2.03 \times 10^{15}/\text{cm}^2\text{sec}$ versus core height at time equal 18 hours and power equal 92 % of full power.
FIGURE 16. Flux in units of $1.42 \times 10^{14}$/cm$^2$sec and xenon in units of $2.03 \times 10^{15}$/cm$^2$sec versus core height at time equal 19 hours and power equal 96% of full power.
FIGURE 17. Reactor power and AO value as a function of time for load-follow pattern A
FIGURE 18. Integrated xenon concentration for load-follow pattern A

to the desired value. During the entire load-follow maneuver, AO stays within 0.5% of the desired value. The spatially integrated xenon concentration shown in Figure 18 increases as power level decreases.

Figure 19 shows the control strategies for load-follow pattern A which were generated by XSCD. These were the control actions used to generate the results shown in Figure 17. The FLCR and the PLCR banks move in opposite directions. Cho and Grossman [37] obtained similar results using optimal
FIGURE 19. The best control strategy corresponding to Figure 17
(c) boron concentration

FIGURE 19. (Continued)
control methods for a similar load-follow pattern. Most of the control used to maintain the desired power distribution during load-follow was provided by the control rods. As the power level changes, coolant inlet temperature and boron concentration were adapted to compensate for the reactivity change due to delayed xenon feedback until the end of the control period. Since the reactor has a negative coolant temperature feedback, the coolant inlet temperature increases with decreasing power level and the temperature decreases with increasing power level as seen in Figure 19(b). Meanwhile, the boron, dissolved in the reactor primary system, is diluted as shown in Figure 19(c). This gives a positive reactivity effect competing with the negative reactivity effect of the coolant inlet temperature. After the power level attains the desired final steady state power (100%), positions of all control variables return to the initial steady state control position.

The other load-follow patterns (pattern B, C, D) presented in Table 6 were also tested by the XSCD expert system. The results obtained are given in Figures 20 through 25. Power and xenon distributions similar to those shown in Figures 12 through 16 were produced during the test of the other load-follow patterns. The results consist of two parts. One is the comparison of the desired power trace
and AO with the current power trace and AO similar to Figure 17. The other graphs show the best control strategy such as Figure 19.

The results of the test performed on pattern B in Table 6 are presented in Figures 20 and 21. The change in desired power level with time is shown as the solid curve in Figure 20. This pattern consists of lowering or raising the power level at a rate of 2 or 4 % per hour. As seen in Figures 20 and 21, the XSCD causes the control system to produce the required power level and AO trace.

The results of the test performed on pattern C are presented in Figures 22 and 23. The change in the desired power level with time is shown as the solid line in Figure 22. This pattern requires a 10 % lowering of power at a rate of 2 % per hour. As seen in Figure 23, the control variables do not change as much as for pattern B. This is because this load pattern requires only a 10 % power decrease, while pattern B requires a 20 % decrease. As seen in Figure 22, the XSCD generates the control steps required to drive the load and AO demands to their respective target bands (±1 %, ±0.5 %).

The results of the test performed on pattern D are presented in Figures 24 and 25. The change in desired power level with time is shown as a solid line in Figure 24. The
FIGURE 20. Reactor power and AO value as a function of time for load-follow pattern B
FIGURE 21. The best control strategy corresponding to Figure 20
FIGURE 21. (Continued)
FIGURE 22. Reactor power and AO value as a function of time for load-follow pattern C
FIGURE 23. The best control strategy corresponding to Figure 22
FIGURE 23. (Continued)
FIGURE 24. Reactor power and AO value as a function of time for load-follow pattern D
FIGURE 25. The best control strategy corresponding to Figure 24
FIGURE 25. (Continued)
final power level is 86 %. As seen in Figure 25, the control variables are changed according to the load demand. Since the final power is a constant 86 %, the control variables do not return to the initial positions as they did when the power was returned to 100 %. The predicted power level and AO follow the desired levels, as seen in Figure 24.

It should be noted that the rules of the XSCD expert system were not altered for the different load-follow patterns. As described in section 6.3, the control inputs were coded and fixed in the rules. The control variables were adjusted at a given time step by the control inputs from a rule corresponding to the existing pattern of power level and shape until both APOERR and AAOERR were within the respective target band. In other words, at a given time step, the control variables were continuously adjusted to produce the target band values of power level and AO.

The XSCD expert system generates good control strategies for load-follow patterns which consist of lowering or raising the power level at a rate of 2 % or more per hour or leveling the power level for a given amount of time.

The execution of the expert system is different from a conventional program in that a conventional program is
executed in one step, but the XSCD expert system queries the user's intention at each step. It accepts commands from the user and causes them to be carried out, one by one.

The simulation of the XSCD expert system took about 40 minutes on the IBM PC-XT. This is acceptable for two reasons. The first is that the expert system is an advisory system where implementation time depends on the capability of the computer hardware. The second reason is that the xenon oscillation is a very slow transient occurring with a period of approximately a day so that reactor operators have enough time to obtain advice from the expert system.
7. CONCLUSIONS AND RECOMMENDATIONS

7.1 Conclusions

The advisory expert system, XSCD, has been developed to illustrate the feasibility of using expert system technology to aid reactor operators in the avoidance of axial xenon oscillations in a PWR. The knowledge base of XSCD is based on production rules. The inference engine uses a forward-chaining inference mechanism.

It was found that the production rules in the form of IF-THEN condition-action pairs were very useful for representing the control knowledge for the load-follow domain. By combining all possible core state variables for power level patterns with all possible core state variables for power shape patterns, 128 patterns of power level and shape were generated. These patterns and their corresponding control actions, all based on the well known constant axial offset control (CAOC) procedure were represented in the knowledge base using IF-THEN production rules. In addition, nine more rules for parameter passing between the XSCD expert system and the simulation core control model, activating graphic programs, creating a POWEROUT.TXT XSCD expert system output file, and display of the control variables were stored in the knowledge base. Thus, the knowledge base consists of 137 rules.
The modularity of the production rules enables each rule to match a separate pattern of power level and shape and to take a corresponding control action. The uniform data structure of the rule base is compact and easy to create, understand, and modify.

Six different core state variables were found to be enough to represent the possible patterns of power level and shape. The core state variable POERR worked well in inferring the direction of FLCRs (downwards or upwards), the necessity of a dilution or boration process, and the direction of coolant inlet temperature (increase or decrease). The sign of PORATE allowed the system to see the power level trend. The APOERR was divided into three quantifying linguistic terms to describe the magnitude of power error; big, medium, and small (target) band. Based on the linguistic term of APOERR with the status of both POERR and PORATE, the APOERR worked well in quantifying the control adjustments of control variables (DELZ1, DELB, and DELT).

The AOERR worked well in inferring the direction of PLCRs (downwards or upwards). The sign of AORATE also allowed the system to infer the trend in power shape shift. Also, the AAOERR was divided into big, medium, and small (target) bands. These were successful in quantifying the
control adjustment of control variable DELZ2. Thus, all possible combinations of the six state variables were found to be adequate to cover all possible patterns of power level and shape.

It was found that the rule learning process was effective to develop a set of unit control input values. The unit control input values depend on the weight values for control variables, which were assigned subjectively. A trial and error strategy using two performance indexes was useful to obtain the maximum unit control input values with which the most accurate control strategy can be achieved.

The forward-chaining inference process was effective in driving the reactor power towards the desired pattern, while maintaining AO within the target band.

As mentioned in Chapter 6, different types of load-follow patterns were tested using XSCD. A suitable control strategy was provided depending on the type of scheduled load pattern. The power and AO predicted by the control strategy closely follow the scheduled load demand within the target error band and the desired power shape is maintained. The results show that XSCD is capable of covering different types of load-follow patterns. This capability, however, depends on the set of unit control input values for the control variables, which was chosen by the rule learning process.
The results obtained by using XSCD were compared with the results of Cho and Grossman's optimal control method [37]. They tested one load-follow pattern which is similar to patterns A and E. The control strategy of XSCD for patterns A and E is in qualitative agreement with that of Cho and Grossman. This demonstrates that XSCD can provide control strategies similar to those obtained using optimal control methods. Furthermore, this shows the potential usefulness of XSCD in specifying control to avoid axial xenon oscillations.

From this research, the following conclusions have been drawn:

1. The solution of the XSCD expert system closely follows the desired load demand and maintains the desired power distribution (essentially constant AO).

2. The XSCD expert system adaptively governs the overall behavior of control variables by recognizing the current patterns, deciding the corresponding control actions, predicting future behavior, and monitoring its execution repeatedly to ensure success.

3. The forward-chaining inference engine is very useful for the advisory control system using system-state pattern-matching.
4. As demonstrated in Figures 17 through 25, the control inputs employed in the rules cover a wide range of load-follow patterns. This is one of the successful achievements of this research.

5. The graphic engine developed can be easily incorporated into the knowledge base and efficiently implemented with the inference engine.

6. An expert system shell like INSIGHT2+ can be used to build an expert system capable of advising reactor operators in the avoidance of slow transient problems such as xenon oscillations. In particular, the capability to interface with external programs is very useful.

In addition, other general conclusions can be drawn from the experiences of this research. First, although expert systems provide a tool for better operational performance, they also contain the expertise of experienced plant personnel. This will insure the utility a continuity in the level of expertise during times of personnel changes. Actually, these expert systems will enable less experienced engineering, operation, and maintenance personnel to perform at a far greater level of expertise than they would otherwise be capable. This tool will become more powerful
with use because the knowledge base of the system will grow with time.

Although the application of AI in the form of expert systems has many potential uses in a nuclear power plant, it also has its limitations. Unlike a human expert, expert systems can not duplicate the intuition and creativity often displayed by the human expert. Also, an expert system can not work effectively outside of its knowledge domain. Specifically, a human expert can draw upon past experiences to solve problems, whereas an expert system can only draw upon the knowledge and rules at hand.

7.2 Recommendations for Future Work

The overall purpose is to implement the expert system technology to the real load-follow operation of a PWR so that the reactor can be operated more safely and economically. This research has demonstrated the feasibility of implementing an expert system to avoid xenon oscillations during a load-follow operation of a PWR.

The XSCD expert system developed in this research was tested using a simulated PWR core control model described by one-group diffusion theory with moderator temperature and xenon-iodine feedbacks [37], which treats a coolant inlet temperature variation as a control variable. It was found
that the coolant inlet temperature variation was an important factor to achieve satisfactory control performance. The coolant inlet temperature might be more appropriately considered as an output variable of the steam generator. It is therefore recommended that the expert system be tested using a more realistic core control code of a PWR which considers a reactor core control model combined with a steam generator control model. In addition, the pressurizer control model in the primary loop would be another part of the control model which might be considered.

The core state variables used in the XSCD expert system are based on the desired power level and AO and the current power level and AO. Thus, the rules used to match a pattern of power level and shape can be directly applied to a more realistic core control model. However, the control inputs used in the XSCD expert system can not be directly used in the other core control model because their values depend on the core control model. Therefore, the XSCD expert system can be used for a different core control model, using corresponding control inputs.

Another way to implement the XSCD expert system is to use a design code of a PWR as a core control model. For example, the CAOC procedure was demonstrated [35] by simulations based on a standard Westinghouse design code,
such as PANDA [77]. Simulating the expert system in the design code, the expert system can advise reactor operators in the load-follow operation by predicting the best control strategy for a scheduled load pattern in advance. However, this requires another set of control inputs different from those for the core control model used in this research.

The design code can be easily interfaced with the expert system by modifying one or more parts of the code and then adding the ASCIIPRM.FOR subroutine to the modified code. Since most of the simulation design codes for load-follow operation utilize two energy-group diffusion theory with the majority of cases performed in one (axial) space dimension, it is possible for such codes to run on a microcomputer without worrying about the memory size of the microcomputer. On the other hand, some codes require more memory than the microcomputer has. However, this kind of problem can be avoided by modifying the structure of the code appropriately to run on the microcomputer.

7.2.1 Implementation of XSCD for a load-follow of a PWR

The best control strategy generated by XSCD can be used either by the reactor operators or by a computer.

Since the xenon oscillations are a very slow transient, occurring once a day, reactor operators have enough time
before deciding the best control strategy for the expected xenon oscillations. For example, the most widely used strategy for the axial xenon oscillations in PWRs is the half-cycle damping method [35,77,78]. The half-cycle damping method is a heuristic control procedure which should be performed by reactor operators during a load-follow operation. Based on the power distribution (or AO value) displayed on the control board, reactor operators take a series of appropriate control actions. However, the control actions depend on the expertise of the reactor operator. The best control strategy can be generated in advance by simulating the expert system using a realistic core control model and can be used by the reactor operator as a control adviser (or procedure) during the scheduled load-follow operation.

A long term goal would be to have the computer actually control the reactor. The best control strategy obtained in advance can be stored in a computer. Using the computer, the control systems can be set up before a scheduled load-follow operation. The computer would then adjust the control systems according to the best control strategy previously determined. Furthermore, current expert system technology allows a knowledge engineer to design an expert system which interacts directly with reactor signals. These
approaches have advantages of inherent reliability of computers and availability under almost all circumstances. In particular, these are important because reactor operators are burdened by stress and emotional factors that have a dramatic effect on performance level. However, use of a system like this requires a number of considerations, namely computer reliability, quality of computer hardware and software, and cost.
8. BIBLIOGRAPHY


24. Teknowledge Inc. 525 University Avenue, Palo Alto, CA 94301.


69. Inference Corporation. 5300 West Century Blvd., Fifth Floor, Los Angeles, CA 90045.

70. Software A&E Inc. 1500 Wilson Blvd., Suite 800, Arlington, VA 22209.

72. Texas Instruments Inc. P.O. Box 809063, Dallas, TX 75380.


75. General Research Corporation, P.O. Box 6770, Santa Barbara, CA 93160.


ACKNOWLEDGEMENTS

I wish to express my deepest gratitude to my major professors Dr. R. A. Danofsky and Dr. B. I. Spinrad, not only for their advice, guidance, and help throughout my studies, but also for their kindness and encouragement during my stay at Iowa State University. Without their advice, the final results could not have been as personally satisfying an investigation.

I am greatly indebted to the Department of Nuclear Engineering for providing financial support.

I am also want to thank Dr. D. M. Roberts, Dr. G. R. Leuke, and Dr. C. T. Wright for their help as committee members. Particularly, I really appreciate Dr. Leuke for his guidance.

Appreciation is also due my fellow graduate students; Mike Winter, Anil Macwan, Ramin Mikail, Uner Colak, and Niyazi Sokmen for their kindness and discussion.

Very special thanks are due to my mother, my wife, Ji-Won and my two daughters, Sun-Young and Moon-Young, and my special friend Sung-Keun and Myung-Hae Park for their kindness.
10. APPENDIX A. SOURCE LIST OF STEADY.FOR PROGRAM

This program is to calculate the steady state flux distribution with the finite difference method. Xenon and coolant temperature effect are included in this model. Initial flux is generated from the model without the effect of xenon and temperature.

```
implicit real*8 (a-h,o-z)
dimension pi0(61),pi1(61),pi2(61),sigc(4),afact(61,61),+
sour(61)
common/svtdc/sigcp,sigcf,vsigf,tp1,tp2,tp5,dif,conver,w
common/fissi/sigf,size
common/iposit/ix(3)
common/cross1/siga,sigc,pfact
common/cross2/conf,boron,sigb,buck,height
common/cross3/mesh,number

vsigf = 0.16d0
tp5 = 0.1385d0
dif = 1.2d0
conver = 1.0d-3
w = 1.3d0
sigf = 0.067d0

vsigf: product of v=2.4 and fission cross section,0.16 cm

siga : fission cross section

bobon: boron concentration ppm

sigb : micro neutron absorption cross section of boron-10

buck : radial bucking (2.405/r)**2

height: reactor height 370 cm

w : acceleration factor for convergence

mesh : even number, 20, 40, 60, 80, or 100; etc

Generating the initial flux

tpl = siga + dif * buck

tp2 = conf * boron * sigb

tp3 = vsigf - tpl - 2.4092d22 * tp2

tp4 = (height ** 2) / dif

mesh = mesh + 1
```
calling flux to generate the initial flux pil

call flux(tp3, tp4, pil, nesh)

calling the iteration routine having xenon and temp effects

sigcp = sigc(2)
sigcf = sigc(4)
do 101 i = 1, nesh
   pil(i) = pfact * pil(i)
   pi0(i) = pil(i) / pfact
101 continue

call steady(nesh, pil, pi2, afact, sour, iter)

writing the xenon, density, temperature
open(8, file='steady.dat', status='unknown')
do 103 i = 1, nesh
   temper = tefect(i, pi2)
   xedat = xenon(i, pi2(i))
   densi = waden(temper)
   degree = 3.0d2 + temper
write(8, 404) pi2(i), xedat, densi, degree, pi0(i)
103 continue

404 format(5(ell.4, 2x))
close(8)

stop
end

block data
double precision siga, sigc(4), pfact, conf, boron, sigb, buck,
   + height
common/cross1/siga, sigc, pfact
common/cross2/conf, boron, sigb, buck, height
common/cross3/mesh, number
common/posit/ix(3)
data siga, sigc, pfact/0.1438d0, 0.d0, 0.65d-3, 0.d0, 0.35d-2,
   + 1.42d14/
data conf, boron, sigb, buck, height/1.67d-6, 518.d0, 750d-24,
subroutine flux(tp3,tp4,pil,nesh)
double precision sigc(4),pil(nesh),a(6,6),aa(5,5),siga,pfact
double precision w(5),u(5,5),v(5,5),rvl(5),b(5),geninv(5,5)
double precision h,alpha,beta,xx,tp3,tp4,x
common/albex,alpha(4),beta(4),x(3)
common/coef/xx(5)
common/posit/ix(3)
common/crossl/siga,sigc,pfact
logical matu,matv

mesh = nesh - 1
mm = 6
nm = 5
m = nm
n = m
matu = .true.
matv = .true.

d = 1.0d0 / float(mesh)

changing the integer position to real position of control rod

do 109 i = 1, 3
   x(i) = ix(i)/100.0
109 continue

do 100 i = 1, 4
   alpha(i) = tp4 * (tp3 - sigc(i))
   beta(i) = dsqrt(dabs(alpha(i)))
100 continue

generating coefficient matrix a(6*6)

do 101 i = 1, 3
   call genera(i,mm,a)
101 continue

generating the aa matrix

do 102 j = 1, n+1
   do 102 i = 1, n
      if(j.eq.6) then
         b(i) = a(i,j)
      endif
102 continue
else
    aa(i,j) = a(i,j)
endif
102 continue

calling svd

call svd(nm,m,n,aa,matu,matv,v,ierr,rvl)

making the inverse matrix to find the solution

geninv is the general inverse matrix

do 103 i = 1, n
do 103 j = 1, n
  geninv(i,j) = 0.0d0
103 continue

do 104 k = 1, n
    if(w(k).lt.1.0d-7) go to 104
    do 105 i = 1, n
do 105 j = 1, n
    geninv(i,j) = geninv(i,j) + v(i,k)*u(j,k)/w(k)
104 continue

calling the solution of the matrix aa

do 106 i = 1, n
  xx(i) = 0.0d0
106 continue

do 107 i = 1, n
do 107 j = 1, n
  xx(i) = geninv(i,j) * b(j) + xx(i)
107 continue

generating the flux distribution

call solut(nesh,h,pil)

return
end

subroutine steady(nesh,pil,pi1,afact,sour,iter)
double precision pil(nesh),pi1(nesh),afact(nesh,nesh)
double precision sour(nesh),sigcp,sigcf,vsigf,sigf,tpl,tp2
double precision dif,conver,w,contol,size,suml,sum2,pmax
double precision differ,alp,xenon,waden,tefect
double precision xedat,densi,temper,tp5,degree
common/svtdc/sigcp,sigcf,vsigf,sigf,tpl,tp2

common/fissi/sigf,size
common/posit/ix(3)
common/cross3/mesh,number
data fact1,sigx/9.46d-4,1.4568044d-18/

size = 3.7d2 / float(mesh)
ipal = (mesh * ix(1)) / 100
iful = (mesh * ix(3)) / 100
ipal = ipal + 1
iful = iful + 1

iteration starting

do 1000 iter = 1, number
  do 100 i = 1, mesh
    if (i.eq.ipal) then
      contol = sigcp
    else
      if (i.eq.iful) then
        contol = sigcf
      else
        contol = 0.d0
      endif
    endif
    temper = tefect(i,pil)
    alp = tp5 + sigx*xenon(i,pil(i)) + contol + tp2*waden(temper) +
         fact1* vsigf * temper
  endif
  continue

  generating tridiagonal coefficients afact(i,j)
  if (i.eq.1) then
    do 97 j = i, i+1
      if (j.eq.i) then
        afact(i,j) = - (2.0d0 + (size**2 / dif) * alp)
      else
        afact(i,j) = 1.0d0
      endif
    97 continue
  else
    if (i.eq.mesh) then
      do 98 j = i-1, i
        if (j.eq.i) then
          afact(i,j) = - (2.0d0 + (size**2 / dif) * alp)
        else
          afact(i,j) = 1.0d0
        endif
      98 continue
    else
      do 99 j = i+1, mesh
        if (j.eq.i) then
          afact(i,j) = - (2.0d0 + (size**2 / dif) * alp)
        else
          afact(i,j) = 1.0d0
        endif
      99 continue
    endif
98    continue
else
do 99  j = i-1, i+1
   if(j.eq.i) then
      afact(i,j) = -(2.0d0 + (size**2 / dif) * alp)
   else
      afact(i,j) = 1.0d0
   endif
99    continue
endif
endif

c    sour(i) = -(size**2/dif)*vsigf*pil(i)
100   continue

c    do 102 i = 1, nesh
       if(i.eq.1.or.i.eq.nesh) then
          pi2(i) = 0.d0
       else
          if(i.eq.mesh) then
             suml = 0.d0
             sum2 = 0.d0
             do 104 j = 1, i-1
                suml = suml + afact(i,j)*pi2(j)
             104 continue
             pi2(i) = (w/afact(i,i))*(sour(i)-suml-sum2)+(1.d0-w)*pil(i)
          else
             suml = 0.d0
             sum2 = 0.d0
             do 105 j = 1, i-1
                suml = suml + afact(i,j)*pi2(j)
             105 continue
             do 106 j = i+1, nesh
                sum2 = sum2 + afact(i,j)*pil(j)
             106 continue
             pi2(i) = (w/afact(i,i))*(sour(i)-suml-sum2)+(1.d0-w)*pil(i)
          endif
       endif
102   continue

c    convergence testing by comparing the max norm of pil and pi2
    with the convergence factor, conver.
pmax = 0.d0
do 107 i = 2, mesh
differ = dabs((pil(i) - pi2(i)) / pil(i))
if(differ.gt.pmax) then
  pmax = differ
endif
107 continue

comparing max norm with the convergence factor

if(pmax.gt.conver) then
  do 108 i = 1, nesh
    pil(i) = pi2(i)
  108 continue
else
  return
endif
1000 continue
return
end

double precision function tefect(i,pil)
double precision pil(i),ytrap,fact2,trap
data fact2/7.9623d-16/

c: factl: corrected coolant temp reactivity coefficient
repeated trapezoidal rule is used to integrated the flux

tytrap = trap(i,pil)
tefect = fact2 * ytrap
return
end

double precision function waden(dat)
double precision dat,dinit,beta
data beta,dinit/3.0d-3,2.4092d22/

c: dinit: coolant inlet number density 0.72g/cm**3
waden = dinit*(1.0d0 - beta * dat)
return
end
double precision function xenon(i,pi)
  double precision rx,ri,pi,sigf,sigx,decayx,stepl,size
  common/fissi/sigf,size
  data rx,ri,decayx,sigx/2.28d-3,6.386d-2,2.092d-5,1.4568044d-18/
  
  stepl = ((rx+ri)*sigf*pi)/(decayx+sigx*pi)
  xenon = stepl
  return
  end

double precision function trap(i,pil)
  double precision pil(i),size,ysum,y,sigf
  common/fissi/sigf,size
  
  ysum = 0.d0
  do 300 j = 1, i
      y = pil(j)
      if(j.eq.1.or.j.eq.i) then
          y = y / 2.d0
          ysum = ysum + y
      else
          ysum = ysum + y
      endif
  300 continue
  trap = ysum * size
  return
  end

subroutine solut(nesh,h,pil)
  implicit real*8 (a-h,o-z)
  dimension pil(nesh)
  common/albex/alpha(4),beta(4),x(3)
  common/coef/xx(5)
  common/iposit/ix(3)
  
  mesh = nesh - 1
  ip1 = (mesh * ix(1)) / 100
  ip2 = (mesh * ix(2)) / 100
  ip3 = (mesh * ix(3)) / 100
  
  do 333 i = 1, 4
     if (alpha(i).gt.0.0d0) then
       if(i.eq.1) then
           start = 0.d0
do 10 j = 1, Ipl
    pil(j) = xx(i) * dsin(beta(i)*start)/beta(i)
    start = start + h
10 continue

start = 0.d0
else
    if(i.eq.2) then
        start = x(i-l)
        do 20 j = ipl+1, ip2
            pil(j) = xx(i)*dcos(beta(i)*start)+xx(i+1)
            *dsin(beta(i)*start)/beta(i)
        start = start + h
20 continue
    start = 0.d0
else
    if(i.eq.3) then
        start = x(i-l)
        do 30 j = ip2+l, ip3
            pil(j) = xx(i+l)*dcos(beta(i)*start)+xx(i+2)
            *dsin(beta(i)*start)/beta(i)
        start = start + h
30 continue
    start = 0.d0
else
    start = x(i-l)
    do 40 j = ip3+l, nesh
        if(j.eq.nesh) then
            pil(j) = 0.0d0
        else
            pil(j) = -dsin(beta(i)*(start-l.d0))/beta(i)
            start = start + h
        endif
40 continue
    endif
else
    if(i.eq.1) then
        start = 0.d0
        do 50 j = 1, ipl
            pil(j) = xx(i) * dsinh(beta(i)*start)/beta(i)
        start = start + h
50 continue
    start = 0.d0
else
    if(i.eq.2) then
        start = x(i-l)
do 60 j = ipl+1, ip2
   pil(j) = xx(i)*dcosh(beta(i)*start) + xx(i+l) + *dsinh(beta(i)*start)/beta(i)
   start = start + h
60 continue

start = 0.d0
else
  if(i.eq.3) then
    start = x(i-l)
  do 70 j = ip2+l, ip3
    pil(j) = xx(i+l)*dcosh(beta(i)*start)+xx(i+2) + *dsinh(beta(i)*start)/beta(i)
    start = start + h
70 continue
  start = 0.d0
else
  start = x(i-l)
  do 80 j = ip3+l, nesh
    if(j.eq.nesh) then
      pil(j) = 0.0d0
    else
      pil(j) = -dsinh(beta(i)*(start-l.d0))/beta(i)
    start = start + h
    endif
80 continue
endif
endif
endif
endif
endif
333 continue

return
end

subroutine genera(i,n,a)
implicit real*8 (a-h,o-z)
dimension a(n,n)
common/albex/alpha(4), beta(4), x(3)

if(alpha(i).gt.0.0d0) then
  if(alpha(i+l).gt.0.0d0) then
    if(i-2) 10,20,30

10 j = 1
   a(i,j) = dsin(beta(i)*x(i))/beta(i)
   a(i,j+1) = -dcos(beta(i+1)*x(i))
   a(i,j+2) = -dsin(beta(i+1)*x(i))/beta(i+1)
a(i+1, j) = - d\cos(\beta(i)*x(i))/\beta(i)
a(i+1, j+1) = - d\sin(\beta(i+1)*x(i))
a(i+1, j+2) = d\cos(\beta(i+1)*x(i))/\beta(i+1)
return

c

j = 2
a(i+1, j) = d\cos(\beta(i)*x(i))
a(i+1, j+1) = d\sin(\beta(i)*x(i))/\beta(i)
a(i+1, j+2) = - d\cos(\beta(i+1)*x(i))
a(i+1, j+3) = - d\sin(\beta(i+1)*x(i))/\beta(i+1)
return

c

j = 4
a(i+2, j) = d\cos(\beta(i)*x(i))
a(i+2, j+1) = - d\cos(\beta(i)*x(i))/\beta(i)
a(i+2, j+2) = - d\sin(\beta(i+1)*x(i))/\beta(i+1)
a(i+2, j+3) = d\cos(\beta(i+1)*x(i))/\beta(i+1)
return

c

else

if(i-2) 11, 21, 31

c

j = 1
a(i, j) = d\sin(\beta(i)*x(i))/\beta(i)
a(i, j+1) = - d\cosh(\beta(i+1)*x(i))
a(i, j+2) = - d\sinh(\beta(i+1)*x(i))/\beta(i+1)
return

c

j = 2
a(i+1, j) = d\cos(\beta(i)*x(i))
a(i+1, j+1) = d\sin(\beta(i)*x(i))/\beta(i)
a(i+1, j+2) = - d\cosh(\beta(i+1)*x(i))
a(i+1, j+3) = - d\sinh(\beta(i+1)*x(i))/\beta(i+1)
return

c

j = 3
a(i+2, j) = d\sin(\beta(i)*x(i))
a(i+2, j+1) = - d\cosh(\beta(i+1)*x(i))
a(i+2, j+2) = d\cos(\beta(i+1)*x(i))/\beta(i+1)
return

c

else
\[ a(i+2,j+2) = \text{dsinh}(\beta(i+1) x(i)) \]
\[ a(i+2,j+3) = \text{dcosh}(\beta(i+1) x(i))/\beta(i+1) \]
\[ \text{return} \]

\[ j = 4 \]
\[ a(i+2,j) = \text{dcos}(\beta(i) x(i)) \]
\[ a(i+2,j+1) = - \text{dsin}(\beta(i+1) x(i))/\beta(i+1) \]
\[ a(i+2,j+2) = - \text{dsinh}(\beta(i+1) (x(i) - 1.0))/\beta(i+1) \]
\[ \text{return} \]

\[ \text{else} \]

\[ \text{if}(\alpha(i+1) > 0.0) \text{ then} \]

\[ i(i-2) 12, 22, 32 \]

\[ j = 1 \]
\[ a(i,j) = \text{dsinh}(\beta(i) x(i))/\beta(i) \]
\[ a(i,j+1) = - \text{dcos}(\beta(i+1) x(i)) \]
\[ a(i,j+2) = - \text{dsin}(\beta(i+1) x(i))/\beta(i+1) \]
\[ a(i+1,j) = - \text{dcosh}(\beta(i) x(i))/\beta(i) \]
\[ a(i+1,j+1) = - \text{dsinh}(\beta(i+1) x(i))/\beta(i+1) \]
\[ a(i+1,j+2) = \text{dcos}(\beta(i+1) x(i))/\beta(i+1) \]
\[ \text{return} \]

\[ j = 2 \]
\[ a(i+1,j) = \text{dcosh}(\beta(i) x(i)) \]
\[ a(i+1,j+1) = \text{dsinh}(\beta(i) x(i))/\beta(i) \]
\[ a(i+1,j+2) = - \text{dcos}(\beta(i+1) x(i)) \]
\[ a(i+1,j+3) = - \text{dsin}(\beta(i+1) x(i))/\beta(i+1) \]
\[ a(i+2,j) = - \text{dsinh}(\beta(i) x(i)) \]
\[ a(i+2,j+1) = - \text{dcosh}(\beta(i) x(i))/\beta(i) \]
\[ a(i+2,j+2) = - \text{dsin}(\beta(i+1) x(i)-1.0))/\beta(i+1) \]
\[ \text{return} \]

\[ j = 4 \]
\[ a(i+2,j) = \text{dcosh}(\beta(i) x(i)) \]
\[ a(i+2,j+1) = \text{dsinh}(\beta(i) x(i))/\beta(i) \]
\[ a(i+2,j+2) = - \text{dsin}(\beta(i+1) x(i)-1.0))/\beta(i+1) \]
\[ \text{return} \]

\[ a(i+3,j) = - \text{dsinh}(\beta(i) x(i)) \]
\[ a(i+3,j+1) = - \text{dcosh}(\beta(i) x(i))/\beta(i) \]
\[
a(i+3,j+2) = d\cos(\beta(i+1)*(x(i)-1.d0))/\beta(i+1)
\]
return

else

if(i-2) 13,23,33

j = 1

a(i,j) = d\sinh(\beta(i)*x(i))/\beta(i)
a(i,j+1) = -d\cosh(\beta(i+1)*x(i))
a(i,j+2) = -d\sinh(\beta(i+1)*x(i))/\beta(i+1)

a(i+1,j) = -d\cosh(\beta(i)*x(i))/\beta(i)
a(i+1,j+1) = d\sinh(\beta(i+1)*x(i))
a(i+1,j+2) = d\cosh(\beta(i+1)*x(i))/\beta(i+1)
return

c

j = 2

a(i+1,j) = d\cosh(\beta(i)*x(i))
a(i+1,j+1) = d\sinh(\beta(i+1)*x(i))/\beta(i)
a(i+1,j+2) = -d\cosh(\beta(i+1)*x(i))
a(i+1,j+3) = -d\sinh(\beta(i+1)*x(i))/\beta(i+1)

return

c

j = 3

a(i+2,j) = d\cosh(\beta(i)*x(i))
a(i+2,j+1) = d\sinh(\beta(i+1)*x(i))/\beta(i)
a(i+2,j+2) = -d\sinh(\beta(i+1)*x(i))/\beta(i+1)

a(i+3,j) = -d\sinh(\beta(i)*x(i))
a(i+3,j+1) = -d\cosh(\beta(i+1)*x(i))/\beta(i)
a(i+3,j+2) = d\cosh(\beta(i+1)*x(i)-1.0d0)/\beta(i+1)

endif
endif
return
end
In order to obtain the matrices A, B, C, D, and E of Eq. (21) of Chapter 5, Eq. (17) are used. Substituting Eq. (19) into Eq. (17) and then forming the inner product with $\psi_j$, the following equations can be obtained. Particularly, the solution of the Helmholtz equation for $i = 1, 2$ are given by Eq. (20b). Thus, Eq. (17a) becomes

$$\frac{1}{\sqrt{V}} \frac{\partial}{\partial t} \left[ (P\int_0^L \psi_1 dz)a_1(t) + (P\int_0^L \psi_2 dz)a_2(t) \right]$$

$$= - D(Q)(P\int_0^L \psi_1 dz)a_1(t) - D(R)(P\int_0^L \psi_2 dz)a_2(t)$$

$$+ \left[ \nu \Sigma_r - \Sigma_z - DB^t + a_w \nu \Sigma_r T_0 \right] \left[ (P\int_0^L \psi_1 dz)a_1(t) + (P\int_0^L \psi_2 dz)a_2(t) \right]$$

$$- \sigma_x(P\int_0^L \psi_1 \psi_1 dz)c_1(t) - \sigma_x(P\int_0^L \psi_2 \psi_2 dz)c_1(t)$$

$$- \sigma_x(P\int_0^L \psi_1 \psi_2 dz)c_2(t) - \sigma_x(P\int_0^L \psi_2 \psi_2 dz)c_2(t)$$

$$- \sigma_x(P\int_0^L \chi_0 \psi_1 \psi_1 dz)a_1(t) - \sigma_x(P\int_0^L \chi_0 \psi_2 \psi_2 dz)a_1(t)$$

$$- \sigma_x(P\int_0^L \chi_0 \psi_1 \psi_2 dz)a_2(t) - \sigma_x(P\int_0^L \chi_0 \psi_2 \psi_2 dz)a_2(t)$$

$$- PP(P\int_0^L \psi_1 \delta(z - z_{10}) \psi_1 dz) \delta z_{10}(t)$$

$$- PP(P\int_0^L \psi_2 \delta(z - z_{10}) \psi_2 dz) \delta z_{10}(t)$$

$$- QQ(P\int_0^L \psi_1 (\delta(z - z_{20}) - \delta(z-z_{20} - 1) \psi_1 dz) \delta z_{20}(t)$$

$$- QQ(P\int_0^L \psi_2 (\delta(z - z_{20}) - \delta(z-z_{20} - 1) \psi_2 dz) \delta z_{20}(t)$$
$$- \text{PP}(\int_0^L H(z - z_{1,0}) \psi_1^2 dz) a_1(t)$$

$$- \text{PP}(\int_0^L H(z - z_{1,0}) \psi_1 \psi_2 dz) a_1(t)$$

$$- \text{QU}\left[\int_0^L (H(z - z_{2,0}) - H(z - z_{2,0} - 1)) \psi_1^2 dz\right] a_1(t)$$

$$- \text{QU}\left[\int_0^L (H(z - z_{2,0}) - H(z - z_{2,0} - 1)) \psi_1 \psi_2 dz\right] a_1(t)$$

$$- \text{PP}(\int_0^L H(z - z_{1,0}) \psi_1 \psi_2 dz) a_1(t)$$

$$- \text{PP}(\int_0^L H(z - z_{1,0}) \psi_2^2 dz) a_1(t)$$

$$- \text{QU}\left[\int_0^L (H(z - z_{2,0}) - H(z - z_{2,0} - 1)) \psi_1 \psi_2 dz\right] a_1(t)$$

$$- \text{QU}\left[\int_0^L (H(z - z_{2,0}) - H(z - z_{2,0} - 1)) \psi_2^2 dz\right] a_1(t)$$

$$- C_b \sigma_B (\int_0^L N_{w_o} \psi_1 \psi_1 dz) \delta B - C_b \sigma_B (\int_0^L N_{w_o} \psi_2 \psi_2 dz) \delta B$$

$$+ C_b \sigma_B (\int_0^L \int_0^Z \psi_1 \psi_1 dz \cdot N_{w_o} \psi_1 \psi_1 dz) a_1(t)$$

$$+ C_b \sigma_B (\int_0^L \int_0^Z \psi_1 \psi_2 dz \cdot N_{w_o} \psi_1 \psi_2 dz) a_1(t)$$

$$+ C_b \sigma_B (\int_0^L \int_0^Z \psi_2 \psi_1 dz \cdot N_{w_o} \psi_1 \psi_1 dz) a_1(t)$$

$$+ C_b \sigma_B (\int_0^L \int_0^Z \psi_2 \psi_2 dz \cdot N_{w_o} \psi_1 \psi_2 dz) a_1(t)$$

$$+ C_b \sigma_B (\int_0^L \int_0^Z \psi_1 \psi_1 dz) \delta T_{1, n}(t)$$

$$+ C_b \sigma_B (\int_0^L \int_0^Z \psi_2 \psi_2 dz) \delta T_{1, n}(t)$$

$$- C_b \sigma_B (\int_0^L N_{w_o} \psi_1 \psi_1 dz) a_1(t) - C_b \sigma_B (\int_0^L N_{w_o} \psi_1 \psi_2 dz) a_1(t)$$

$$- C_b \sigma_B (\int_0^L N_{w_o} \psi_2 \psi_2 dz) a_2(t) - C_b \sigma_B (\int_0^L N_{w_o} \psi_2 \psi_1 dz) a_2(t)$$
Eq. (17b) of Chapter 5 also becomes

\[
\frac{3}{3t} [(P_{\int_0^L, \psi_1 dz})b_1(t) + (P_{\int_0^L, \psi_2 dz})b_2(t)]
\]

\[
= \gamma_x \Sigma \left[ (P_{\int_0^L, \psi_1 dz})a_1(t) + (P_{\int_0^L, \psi_2 dz})a_2(t) \right]
\]

\[- \lambda_x \left[ (P_{\int_0^L, \psi_1 dz})b_1(t) + (P_{\int_0^L, \psi_2 dz})b_2(t) \right]
\]

Eq. (17c) of Chapter 5 becomes

\[
\frac{3}{3t} [(P_{\int_0^L, \psi_1 dz})c_1(t) + (P_{\int_0^L, \psi_2 dz})c_2(t)]
\]

\[
= \gamma_x \Sigma \left[ (P_{\int_0^L, \psi_1 dz})a_1(t) + (P_{\int_0^L, \psi_2 dz})a_2(t) \right]
\]

\[+ \lambda_x \left[ (P_{\int_0^L, \psi_1 dz})b_1(t) + (P_{\int_0^L, \psi_2 dz})b_2(t) \right]
\]

\[- \lambda_x \left[ (P_{\int_0^L, \psi_1 dz})c_1(t) + (P_{\int_0^L, \psi_2 dz})c_2(t) \right]
\]
\[-\sigma_x \left[ (P \int_0^L \psi_1 dz) c_1(t) + (P \int_0^L \psi_1 \psi_2 dz) c_1(t) \right] \\
+ (P \int_0^L \psi_1 \psi_2 dz) c_2(t) + (P \int_0^L \psi_1 \psi_2 dz) c_2(t) \right] \\
- \sigma_x \left[ (P \int_0^L X_0 \psi_1 \psi_2 dz) a_1(t) + (P \int_0^L X_0 \psi_1 \psi_2 dz) a_1(t) \right] \\
+ (P \int_0^L X_0 \psi_1 \psi_2 dz) a_2(t) + (P \int_0^L X_0 \psi_2 dz) a_2(t) \right] \]

where

\[ P = \frac{2}{L} \]

\[ Q = \frac{\pi^2}{L^2} \]

\[ R = \frac{4\pi^2}{L^2} \]

\[ PP = \frac{A_x \Sigma_{c,\gamma}}{2\pi J_0} \sum_{i=1}^{n} J_0(B_x r_i) \]

\[ QQ = \frac{A_x \Sigma_{c,\beta}}{2\pi J_0} \sum_{j=1}^{n} J_0(B_x r_j) \]

\[ RB = \frac{2\pi e \Sigma_{c,\beta} \bar{J}_0}{WC_p} \]

\[ RR = \frac{2\pi e \Sigma_{c,\beta} \bar{J}_0}{WC_p} \]

\[ \bar{J}_0 = \int_0^R r J_0(B_x r) dr \]

\[ \psi_o = \text{steady state neutron flux in the axial direction.} \]

\[ X_o = \text{steady state xenon concentration in the axial direction.} \]
direction.

\( N_{w_o} = \text{steady state coolant density at z direction.} \)

\( T_{c_o} = \text{steady state temperature distribution at z direction.} \)

Rewriting above equations in the forms of Eq. (21) of Chapter 5, we can obtain element of each matrix: \([A]\), \([B]\), \([C]\), \([D]\), and \([E]\).

(1) \([A]\) matrix

\[
A_{11} = - D(Q) + [\nu \Sigma_r - \Sigma_a - DB^2 + a_m \nu \Sigma_r T_c] \\
- \sigma_x \left( P \int_{z_0}^{z_l} \psi_1^2 dz \right) \\
- PP \left( P \int_{z_{10}}^{z_{20}} \psi_1^2 dz \right) - QQ \left( P \int_{z_{10}}^{z_{20}} \psi_1^2 dz \right) \\
+ C_b \sigma_b B_o \left[ (2/\pi) \int_0^l (1 - \cos(\pi z/L)) N_{w_o} \psi_0 \psi_1 dz \right] \\
- C_b \sigma_b B_o \left( P \int_{z_0}^{z_l} N_{w_o} \psi_1^2 dz \right) \\
- a_m \nu \Sigma_r \left( P \int_{z_0}^{z_l} \psi_1 \psi_2 dz \right) \\
+ C_b \sigma_b B_o \left[ (1/\pi) \int_0^l (1 - \cos(2\pi z/L)) N_{w_o} \psi_0 \psi_1 dz \right] \\
- C_b \sigma_b B_o \left( P \int_{z_0}^{z_l} N_{w_o} \psi_1 \psi_2 dz \right)
\]
\[ A_{11} = - \sigma_z (P \int_0^L X_0 \psi_1 \psi_2 \, dz) - (PP) (P \int_{z_{10}}^L \psi_1 \psi_2 \, dz) \]

\[ - (QQ) (P \int_{z_{20}}^{z_{20}+\gamma} \psi_1 \psi_2 \, dz) \]

\[ + C_b \sigma_z B_0 (RB) [(2/\pi) \int_0^L (1 - \cos(\pi z/L)) N_{\psi_0} \psi_1 \psi_2 \, dz] \]

\[ - C_b \sigma_z B_0 (P \int_0^L N_{\psi_0} \psi_1 \psi_2 \, dz) \]

\[ - a_m \nu \Sigma_r (RR) [\nu \Sigma_r - \Sigma_s - DB_z^i + a_m \nu \Sigma_r T_0] \]

\[ - a_m \nu \Sigma_r (P \int_0^L T_{\psi_2} \, dz) \]

\[ - \sigma_x (P \int_0^L X_0 \psi_2 \, dz) \]

\[ - PP (P \int_{z_{10}}^L \psi_2 \, dz) - QQ (P \int_{z_{20}}^{z_{20}+\gamma} \psi_2 \, dz) \]

\[ + C_b \sigma_z B_0 (RB) [(1/\pi) \int_0^L (1 - \cos(2\pi z/L)) N_{\psi_0} \psi_2 \, dz] \]

\[ - C_b \sigma_z B_0 (P \int_0^L N_{\psi_0} \psi_2 \, dz) \]

\[ - a_m \nu \Sigma_r (RR) [(1/\pi) \int_0^L (1 - \cos(2\pi z/L)) \psi_0 \psi_2 \, dz] \]

\[ - a_m \nu \Sigma_r (P \int_0^L T_{\psi_2} \, dz) \]

(2) [B] matrix
\( B_{11} = 0.0 \)
\( B_{12} = 0.0 \)
\( B_{13} = -\sigma_x \left( \int_0^L \psi_0 \psi_1^2 \, dz \right) \)
\( B_{14} = -\sigma_x \left( \int_0^L \psi_0 \psi_1 \psi_2 \, dz \right) \)
\( B_{21} = 0.0 \)
\( B_{22} = 0.0 \)
\( B_{23} = -\sigma_x \left( \int_0^L \psi_0 \psi_1 \psi_2 \, dz \right) \)
\( B_{24} = -\sigma_x \left( \int_0^L \psi_0 \psi_2^2 \, dz \right) \)

\((3)\) \([C]\) matrix

\( C_{11} = (PP) \left[ \psi_0(z_{1o}) \sin(\pi z_{1o}/L) \right] \)
\( C_{12} = (QQ) \psi_0(z_{2o}) \sin(\pi z_{2o}/L) \)
\( \quad - \psi_0(z_{2o} + 1) \sin(\pi(z_{2o} + 1)/L) \)

\( C_{13} = -C_o \sigma_3 \left( \int_0^L N_{co} \psi_0 \psi_1 \, dz \right) \)
\( C_{14} = C_o \sigma_3 B_o \beta \left( \int_0^L N_{co} \psi_0 \psi_1 \, dz \right) - a_m \nu \Xi_c \left( \int_0^L \psi_0 \psi_1 \, dz \right) \)
\( C_{11} = (PP) \left[ \psi_0(z_{1o}) \sin(2\pi z_{1o}/L) \right] \)
\( C_{12} = (QQ) \psi_0(z_{2o}) \sin(2\pi z_{2o}/L) \)
\[ - \psi_o(z_{1o} + 1)\sin(2\pi(z_{2o} + 1)/L) \]

\[ C_{13} = -C_b\sigma_b(\mathcal{P}\int_0^L \psi_o^2 \psi_2 \, dz) \]

\[ C_{14} = C_b\sigma_B\beta(\mathcal{P}\int_0^L \psi_o^2 \psi_2 \, dz) - a_n\nu E_r(\mathcal{P}\int_0^L \psi_o \psi_2 \, dz) \]

(4) [D] matrix

\[ D_{11} = -\lambda_r \]

\[ D_{12} = 0.0 \]

\[ D_{13} = 0.0 \]

\[ D_{14} = 0.0 \]

\[ D_{21} = 0.0 \]

\[ D_{22} = -\lambda_r \]

\[ D_{23} = 0.0 \]

\[ D_{24} = 0.0 \]

\[ D_{31} = \lambda_r \]

\[ D_{32} = 0.0 \]

\[ D_{33} = -\lambda_x - \sigma_x(\mathcal{P}\int_0^L \psi_o^2 \psi_2 \, dz) \]

\[ D_{34} = -\sigma_x(\mathcal{P}\int_0^L \psi_o \psi_1 \psi_2 \, dz) \]
\[ D_{41} = 0.0 \]
\[ D_{42} = \lambda_1 \]
\[ D_{43} = -\sigma_x (P \int_0^L \psi_0 \psi_1 \psi_2 dz) \]
\[ D_{44} = -\lambda_x - \sigma_x (P \int_0^L \psi_1 \psi_2 \psi_2 dz) \]

(5) \([E]\) matrix

\[ E_{11} = \gamma_{\xi} \Xi_\xi \]
\[ E_{12} = 0.0 \]
\[ E_{21} = 0.0 \]
\[ E_{22} = \gamma_{\xi} \Xi_\xi \]
\[ E_{31} = \gamma_x - \sigma_x (P \int_0^L \psi_0 \psi_1 \psi_2 dz) \]
\[ E_{32} = -\sigma_x (P \int_0^L \psi_0 \psi_1 \psi_2 dz) \]
\[ E_{41} = -\sigma_x (P \int_0^L \psi_1 \psi_2 \psi_2 dz) \]
\[ E_{42} = \gamma_x - \sigma_x (P \int_0^L \psi_1 \psi_2 \psi_2 dz) \]
This is main routine performing the unsteady state power simulation.

```
implicit real*8 (a-h,o-z)
dimension pi2(61),xe(61),wden(61),tem(61),ix(3),y(6)
dimension pi2t(61),xet(61)
integer Iunit,NmParm,NmRead
character*20 FilNam
common/Param/Iunit,NmParm,NmRead
common/CharC/FilNam
common/tmatix/A(2,2),B(2,4),C(2,4),D(4,4),E(4,2)
common/trans/delzl,delz2,delb,delt
common/stey/pi2,xe,wden,tem
common/trape/size,nesh
common/defa/factl,fact2,sigx,beta,pi,height
common/cross3/sigf,conver,w,ix,mesh
common/tnorm/Ntime

lunit = 10
FilNam = 'PRED.DAT'

nesh = mesh + 1
size = height / float(mesh)

Reading system variables from the knowledge base, where
Ntime:time step, Nstep:Desired power step, Prpowr:previous power, Prao:previous axial offset, y(1)..y(6):Initial values

Call OpenF
Call ReadI(Ntime)
Call ReadI(npoiit)
Call ReadI(Nstep)
Call ReadR(Prpowr)
Call ReadR(Prao)

Receiving control action variables from knowledge base
delzl:full length control rod, delz2:part length control rod
delb:boron concentration, delt:coolant inlet temperature

Call ReadR(delzl)
Call ReadR(delz2)
Call ReadR(delb)
Call ReadR(delt)

making data for control values
open(l,file='control.out',status='unknown',access='direct',
```
+ form='formatted', recl=45)
write(1,600,rec=Ntime) Ntime, delz1, delz2, delb, delt
600 format(i2, lx, 4(f8.3, lx))
if (Ntime.eq.1) then
reading the steady state flux, xenon, density and temp
open(5, file='steady.dat', status='unknown')
i = 1
400 read(5, 404, end=500) pi2(i), xe(i), wden(i), tem(i)
404 format(4(e1.4, 2x))
i = i + 1
go to 400
500 close(5)

Creating data file for the initial value of y(1) .. y(6)
call ylwrit(Ntime, yl)

Cpower: current power, Prpower: previous power, Axs: steady state axial offset, Powers: steady state power, Xenons: steady state xenon concentration

open(7, file='Data1.out', status='unknown')
Axs = axoff(pi2)
Cao = Axs
Powers = trapp(nesh, pi2)
Cpower = (Powers/Powers) * 100.0
Xenons = trapx(nesh, xe)
Cxenon = (Xenons/Xenons) * 100.0
write(7, *) Powers, Axs, Xenons
close(7)

Generating A, B, C, D, E matrix
open(8, file='Data2.out', status='unknown')
call genea(A)
call geneb(B)
call genec(C)
called ned(D)
called genee(E)
writing these matrix to the file data2.out file=8

do 83 i = 1, 2
   write(8,'*') (A(i,j),j=1,2)
   write(8,'*') (B(i,j),j=1,4)
   write(8,'*') (C(i,j),j=1,4)
83 continue

do 84 i = 1, 4
   write(8,'*') (D(i,j),j=1,4)
   write(8,'*') (E(i,j),j=1,2)
84 continue

close(8)

Define the power error, poerr and its magnitude, apoerr
desired power, depow, current power, Cpower
Desire is function program to find out desired power trace.
Poweri is the initial reactor power 100 %

```
Poweri = 100.0
depow = Desire(Nstep)
poerr = ((depow - Cpower)/depow) * 100.d0
apoerr = dabs(poerr)
porate = 0.0
aoerr = AxS - Cao
aaoerr = dabs(aoerr)
aorate = 0.0
```

sending these state variables to the knowledge base

Call ResetF(13)
   Call writeI(Ntime)
   Call writeI(Nstep)
   Call writeR(depow)
   Call writeR(Cpower)
   Call writeR(Cxenon)
   Call writeR(Axs)
   Call writeR(Cao)
   Call writeR(poerr)
   Call writeR(apoerr)
   Call writeR(porate)
   Call writeR(aoerr)
   Call writeR(aaoerr)
   Call writeR(aorate)
   Call CloseF

making data file for the power and axial offset
open(2, file='poao.out', status='unknown', access='direct', + form='formatted', recl=55)
write(2, 601, rec=Ntime) Ntime, Poweri, Cpower, Cxenon, Axs, Cao
601 format(i2, 1x, 5(f8.3, 1x))
else

Retrieving steady state power, xenon, axial offset value, and
coefficient matrix A, B, C, D, E from spowl and spow2

    call spowl(Powers, Axs, Xenons)
call spow2

Reading steady state flux

open(5, file='steady.dat', status='unknown')
do 88 i = 1, nesh
    read(5, 405) pi2(i), xe(i)
88 continue
405 format(2(ell.4, 2x))
call yread for initial value of y
call ylread(yl)
call flux to calculate y(1)....y(6) at each time step
call flux(yl)
z = 0.d0
do 190 i = 1, nesh
    if(i.eq.nesh) then
        delpi = 0.d0
delxe = 0.d0
    else
        delpi = yl(1)*dsin(pi*z/height) +
               yl(2)*dsin(2.d0*pi*z/height)
delxe = yl(5)*dsin(pi*z/height) +
               yl(6)*dsin(2.d0*pi*z/height)
delpi = delpi / 1.42d14
delxe = delxe / 2.03d15
    endif
stepi = pi2(i) / 1.42d14
stexe = xe(i) / 2.03d15
pi2t(i) = stepi + delpi
xet(i) = stexe + delxe
z = z + size
Writing transient flux and xenon value at time t

call neutot(pi2t,nesh)
call xetot(xet,nesh)

where 1.42d14 and 2.03d15 are unit of flux and xenon

current power and axial offset prediction

Cao = axoff(pi2t)
Cpower = (trap(nesh,pi2t)/Powers)*100.d0
Cxenon = (trap(nesh,xet)/Xenons)*100.d0

desired power depow, power rate porate, AO error aoerr
magnitude of AO aoerr, AO rate aorate

depow = Desire(Nstep)
if(depow.ne.0.d0) then
  poerr = ((depow-Cpower)/depow)*100.d0
  apoerr = dabs(poerr)
  porate = Cpower - Prpowr
  aoerr = Axs - Cao
  aaoerr = dabs(aoerr)
  aorate = Cao - Prao
endif

making data file for the power and axial offset

open(2,file='poao.out',status='unknown',access='direct',
     form='formatted',recl=55)
write(2,601,rec=Ntime) Ntime,depow,Cpower,Cxenon,Axs,Cao
else
  poerr = 0.0
  apoerr = 0.0
  aoerr = 0.0
  aaoerr = 0.0
  aorate = 0.0
  porate = 0.0
endif

sending system state variables to the knowledge base
Call ResetF(13)

Call writeI(Ntime)
Call writeI(Nstep)
Call writeR(depow)
Call writeR(Cpower)
Call writeR(Cxenon)
Call writeR(Axs)
Call writeR(Cao)
Call writeR(poerr)
Call writeR(apoerr)
Call writeR(porate)
Call writeR(aoerr)
Call writeR(aorate)
Call CloseF

endif

stop
end

block data
double precision vsigf, siga, sigc, pfact, conf, boron, sigb, dif + , buck, sigf, conver, w, pi, height, fact1, fact2, + sigx, beta, decai, decax, rx, ri
common/defa/fact1, fact2, sigx, beta, pi, height
common/cross1/vsigf, siga, sigc(4), pfact
common/cross2/conf, boron, sigb, dif, buck, decai, decax, rx, ri
common/cross3/sigf, conver, w, ix(3), mesh

data vsigf, siga, sigc, pfact/0.16d0, 0.1438d0, 0.d0, 0.650d-3, 0.d0, +0.35d-2, 1.42d14/ 
data conf, boron, sigb, dif, buck, decai, decax, rx, ri/1.67d-6, 518.d0 +, 750.d-24, 1.2d0, 2.0014d-4, 2.875d-5, 2.092d-5, 2.28d-5, 3.e3d-3, 3.141592d0, 3.7d2/
data fact1, fact2, sigx, beta, pi, height/9.46d-4, 8.4936d-16, +1.4566044d-18, 3.0d-3, 3.141592d0, 3.7d2/
end

subroutine spowl(Powers, Axs, Xenons)
double precision Powers, Axs, Xenons

open(7, file='Data1.out', status='unknown')
    read(7, *) Powers, Axs, Xenons
close(7)
return
end

subroutine spow2
double precision A, B, C, D, E
common/tmatix/A(2, 2), B(2, 4), C(2, 4), D(4, 4), E(4, 2)

open(8, file='Data2.out', status='unknown')

do 83 i = 1, 2
read(8,*) (A(i,j),j=1,2)
read(8,*) (B(i,j),j=1,4)
read(8,*) (C(i,j),j=1,4)
83 continue

do 84 i = 1, 4
    read(8,*) (D(i,j),j=1,4)
    read(8,*) (E(i,j),j=1,2)
84 continue

return
end

double precision function Desire(Nstep)
double precision dp

open(9,file='dpower.dat',status='unknown')

do 100 i = 1, 100
    read(9,*,end=200) jj, dp
    if(jj.eq.Nstep) then
        Desire = dp
        return
    endif
100 continue
200    write(*,*) ' Error in reading the desired power'

return
end

subroutine ylwrit(n,y)
double precision y(6)

open(12,file='initial.out',status='unknown')

if(n.eq.1) then
    do 100 i = 1, 6
        y(i) = 0.d0
    100 continue
    write(12,*) (y(i),i=1,6)
else
    write(12,*) (y(i),i=1,6)
endif
close(12)
return
entry ylread(y)

open(12, file='initial.out', status='unknown')
    read(12, *) (y(i), i=1, 6)
close(12)
c
return
c
end
c
c
subroutine neutot(a, nesh)
double precision a(61)
c
open(18, file='trans1.out', status='unknown')

write(18, *) nesh
    do 100 i = 1, nesh
        write(18, *) i, a(i)
    100 continue
close(18)
c
return
c
end
c
c
entry xetot(a, nesh)

open(19, file='trans2.out', status='unknown')

write(19, *) nesh
    do 150 i = 1, nesh
        write(19, *) i, a(i)
    150 continue
close(19)
c
return
c
end

c
********************************************************************************
c*
Program : FORTRAN Library for ASCII parameter passing to I2+
*c
File Name : ASCIIPRM.FOR
*v
Version : 1.1
*c
Compiler : IBM FORTRAN & PROFORT COMPILERS
*v
Comments : This is the library of FORTRAN subroutines that
provide all the utilities for passing parameter
data between a FORTRAN program and INSIGHT2+
The file ASCIIPRM must be linked with the FORTRAN
program as follows:

    link progaSCIIPRM;

These routines provide parameter passing via disk file. The following statements must be included in your main FORTRAN program prior to any routine calls in order to access the routines:

    Character*20 FilNam
    Common /Param/ Iunit,NmParm,NmRead
    Common /CharC/ FilNam
    Iunit = xx  (unit number)
    FilNam = 'name'  (must be same name as in I2+ knowledge base)

where

Iunit = unit number of parameter file
FilNam = parameter file name (may include full DOS path)
NmParm = of parameters to be received
          (only of use to routines)
NmRead = parameters read so far
          (only of use to routines)

Routines : File Handling

Reading Parameters

    OpenF
    ResetF (NmRetr)
    CloseF

    ReadI (Value)  integer
    ReadR (Value)  real
    ReadC (Value)  character
    ReadB (Value)  boolean
    ReadS (Value)  string

Writing Parameters

    WriteI (Value)  integer
    WriteR (Value)  real
    WriteC (Value)  character
    WriteB (Value)  boolean
    WriteS (Value)  string

Subroutine Messag (MessNo)

This routine will display error messages related to
C* the parameter passing
C*
C******************************************************,******************************************************
C
Integer Iunit,NmParm,NmRead
Character*20 FilNam
Common /Param/ Iunit,NmParm,NmRead
Common /CharC/ FilNam
If (MessNo .EQ. 0) then
  Write ("*,") 'Parameter file empty'
Elseif (MessNo .EQ. 1) then
  Write ("*,") 'I/O Error upon Read'
Elseif (MessNo .EQ. 2) then
  Write ("*,") 'Attempted to read more parameters than passed'
Elseif (MessNo .EQ. 3) then
  Write ("*,") 'Parameter type mismatch'
Elseif (MessNo .EQ. 4) then
  Write ("*,") 'Character or Boolean not found'
Endif
Pause
Return
End

C

Subroutine OpenF
C
C******************************************************,******************************************************
C*
C** This routine opens access to a specified disk file expected to contain parameter data. Also the following variables are set: NmParm NmRead
C*
C******************************************************,******************************************************
C
Integer Iunit,NmParm,NmRead
Character*20 FilNam
Common /Param/ Iunit,NmParm,NmRead
Common /CharC/ FilNam
NmRead = 0
Open (Iunit,File=FilNam)
Read (Iunit,*,End=500,Err=600) NmParm
Return
500 Call Messag (0)
Return
600 Call Messag (1)
Return
End
C
Subroutine ResetF (NmRetr)

This routine resets a disk-based parameter file to the beginning of the file so that return data can be sent. The number of return parameters is also written at the top of the file (this is the routine argument).

Integer form

Integer Iunit, NmParm, NmRead
Character*20 FilNam
Common /Param/ Iunit, NmParm, NmRead
Common /CharC/ FilNam
Open (Iunit, File=FilNam, Status='OLD')
Close (Iunit, Status='DELETE')
Open (Iunit, File=FilNam, Status='NEW')
If (NmRetr .LT. 10) then
  Assign 901 to form
Elseif (NmRetr .LT. 100) then
  Assign 902 to form
Elseif (NmRetr .LT. 1000) then
  Assign 903 to form
Else
  Assign 904 to form
Endif
Write (Iunit, form) NmRetr
901 Format (IX, II)
902 Format (IX, 12)
903 Format (IX, 13)
904 Format (IX, 14)
Return
End

Subroutine CloseF

This routine closes a disk-based parameter file.

Integer Iunit, NmParm, NmRead
Character*20 FilNam
Common /Param/ Iunit, NmParm, NmRead
Common /CharC/ FilNam
Subroutine ReadI (Value)

C**********************************************************************
C*                                                                     *
C*     This routine reads an integer from the parameter data file.     *
C*                                                                     *
C**********************************************************************

Integer Value
Character Type
Integer Iunit,NmParm,NmRead
Character*20 FilNam
Common /Param/ Iunit,NmParm,NmRead
Common /CharC/ FilNam
NmRead = NmRead + 1
If (NmRead .GT. NmParm) then
   Call Messag (2)
Else
   Read (Iunit,901,End=500,Err=600) Type
   If (Type .NE. 'N') then
      Call Messag (3)
   Else
      BackSpace Iunit
      Read (Iunit,902,End=500,Err=600) Type,Rval
      Value = Int(Rval)
   Endif
Endif
500 Call Messag (0)
Return
600 Call Messag (1)
901 Format (A)
902 Format (A,F16.7)
Return
End

Subroutine ReadR (Value)

C**********************************************************************
C*                                                                     *
C*     This routine reads a double precision real number from          *
C*     the parameter data file.                                       *
C**********************************************************************

Close (Iunit)
Return
End

Subroutine ReadR (Value)
Character Type
Integer Iunit,NmParm,NmRead
double precision Value
Character*20 FilNam
Common /Param/ Iunit,NmParm,NmRead
Common /CharC/ FilNam
NmRead = NmRead + 1
If (NmRead .GT. NmParm) then
   Call Messag (2)
Else
   Read (Iunit,901,End=500,Err=600) Type
   If (Type .NE. 'N') then
      Call Messag (3)
   Else
      BackSpace Iunit
      Read (Iunit,902,End=500,Err=600) Type,Value
   Endif
endif
Endif
Return
500 Call Messag (0)
Return
600 Call Messag (1)
901 Format (A)
902 Format (A,F16.7)
Return
End

Subroutine Writel (Value)
C
C This routine writes an integer to the parameter data file.
C**************************************************************************************
C Real Rval
Integer Value
Integer Iunit,NmParm,NmRead
Character*20 FilNam
Common /Param/ Iunit,NmParm,NmRead
Common /CharC/ FilNam
Rval = Float(Value)
Write (Iunit,991) Rval
991 Format ('N ',E13.6)
Return
End

C
Subroutine WriteR (Value)

This routine writes a double precision real number to the parameter data file.

double precision Value
Integer Iunit,NmParm,NmRead
Character*20 FilNam
Common /Param/ Iunit,NmParm,NmRead
Common /CharC/ FilNam
Write (Iunit,991) Value

991 Format ('N ',E13.6)
Return
double precision function trapl(i,j)
implicit real*8(a-h,o-z)
common/stey/pi2(61),xe(61),wden(61),tem(61)
common/defa/factl,fact2,sigx,beta,pi,height
common/trape/size,nesh
common/cross3/sigf,conver,w,ix(3),mesh
common/cross1/vsigf,siga,sigc(4),pfact
common/cross2/conf,boron,sigb,dif,buck,decai,decax,rx,ri
c use the repeated trapezoidal rule for the integral.
c
ysum = 0.d0
z = 0.d0
c
do 300 k = 1, nesh
   y = xe(k)*dsin(i*pi*z/height)*dsin(j*pi*z/height)
   if(k.eq.1.or.k.eq.nesh) then
      y = y / 2.d0
      ysum = ysum + y
   else
      ysum = ysum + y
   endif
   z = z + size
300 continue
trapl = ysum * size
return
c
tenray trap2(i,j,iful)
c
iful = iful + 1
ysum = 0.d0
z = ifull * size
c
do 301 k = ifull, nesh
      y = dsin(i*pi*z/height)*dsin(j*pi*z/height)
      if(k.eq.ifull.or.k.eq.nesh) then
         y = y / 2.d0
         ysum = ysum + y
      else
         ysum = ysum + y
      endif
      z = z + size
 301 continue
      trap2 = ysum * size
      return

      entry trap3(i,j)
      ysum = 0.d0
      ipl = (mesh * ix(1)) / 100
      ip2 = (mesh * ix(2)) / 100
      ipl = ipl + 1
      ip2 = ip2 + 1
      z = ipl * size
      do 302 k = ipl, ip2
         y = dsin(i*pi*z/height)*dsin(j*pi*z/height)
         if(k.eq.ipl.or.k.eq.ip2) then
            y = y /2.d0
            ysum = ysum + y
         else
            ysum = ysum + y
         endif
         z = z + size
 302 continue
      trap3 = ysum * size
      return

      entry trap4(i,j)
      ysum = 0.d0
      z = 0.d0
      do 303 k = 1, nesh
         y = pi2(k)*wden(k)*(1.d0-dcos(j*pi*z/height)) * 
            dsin(i*pi*z/height)
         if(k.eq.1.or.k.eq.nesh) then
            y = y / 2.d0
            ysum = ysum + y
         else
            ysum = ysum + y
         endif
         z = z + size
303 continue
   trap4 = ysum * size
   return
   
   entry trap5(i,j)
   
   ysum = 0.d0
   z = 0.d0
   
   do 304 k = 1, nesh
     y = wden(k)*dsin(i*pi*z/height)*dsin(j*pi*z/height)
     if(k.eq.1.or.k.eq.nesh) then
       y = y / 2.d0
       ysum = ysum + y
     else
       ysum = ysum + y
     endif
     z = z + size
   304 continue
   trap5 = ysum * size
   return
   
   entry trap6(i,j)
   
   ysum = 0.d0
   z = 0.d0
   
   do 305 k = 1, nesh
     y = pi2(k)*(l.d0-dcos(j*pi*z/height))*dsin(i*pi*z/height)
     if(k.eq.1.or.k.eq.nesh) then
       y = y / 2.d0
       ysum = ysum + y
     else
       ysum = ysum + y
     endif
     z = z + size
   305 continue
   trap6 = ysum * size
   return
   
   entry trap7(i,j)
   
   ysum = 0.d0
   z = 0.d0
   
   do 306 k = 1, nesh
     y = tem(k)*dsin(i*pi*z/height)*dsin(j*pi*z/height)
     if(k.eq.1.or.k.eq.nesh) then
       y = y / 2.d0
       ysum = ysum + y
     endif
     z = z + size
   306 continue
   trap7 = ysum * size
   return
else
    ysum = ysum + y
endif
z = z + size
306 continue
trap7 = ysum * size
return

c
entry para(i)
c
reft = 3.0d2
tp = siga + dif * buck
temp = -dif * ((i*pi)/height) ** 2
para = temp + (vsigf - tp + (factl*vsigf*reft))
return
c
entry trap8(i,j)
c
ysum = 0.d0
z = 0.d0
c
do 307 k = 1, nesh
    y = pi2(k)*dsin(i*pi*z/height)*dsin(j*pi*z/height)
if(k.eq.1.or.k.eq.nesh) then
    y = y / 2.d0
    ysum = ysum + y
else
    ysum = ysum + y
endif
z = z + size
307 continue
trap8 = ysum * size
return
c
entry trap9(i)
c
ysum = 0.d0
z = 0.d0
c
do 308 k = 1, nesh
    y = pi2(k)*wden(k)*dsin(i*pi*z/height)
if(k.eq.1.or.k.eq.nesh) then
    y = y / 2.d0
    ysum = ysum + y
else
    ysum = ysum + y
endif
z = z + size
308 continue
trap9 = ysum * size
return

entry tral0(i)

ysum = 0.d0
z = 0.d0

d0 309 k = 1, nesh
y = pi2(k)*dsin(i*pi*z/height)
if(k.eq.1.or.k.eq.nesh) then
  y = y / 2.d0
  ysum = ysum + y
else
  ysum = ysum + y
endif
z = z + size
309 continue
tral0 = ysum * size
return
end

C subroutine genea(a)
implicit real*8 (a-h,o-z)
dimension a(2,2),b(2,4),c(2,4),d(4,4),e(4,2)
common/trape/size,nesh
common/stey/pi2(61),xe(61),wden(61),tem(61)
common/defa/fact1,fact2,sigx,beta,pi,height
common/cross1/vsigf,siga,sigc(4),pfact
common/cross2/conf,boron,sigb,dif,buck,decai,decax,rx,ri
common/cross3/sigf,conver,w,ix(3),mesh

tp2 = conf * boron * sigb
av = 2.0d0 / height
bv = tp2 * fact2 * beta
iful = (mesh * ix(3)) / 100

d0 100 i = 1, 2
do 100 j = 1, 2
  if(i.eq.j) then
    coef = para(i)
  else
    coef = 0.d0
  endif
if(j.eq.1) then
  m = 2
else
m = 1
endif

c para is double precision function to calculate a(i,j)
c
part1 = coef - sigx * av * trap1(i,j)
part2 = part1 - sigc(4) * av * trap2(i,j)
part2 = part2 - sigc(2) * av * trap3(i,j)
c
part2 = bv * (m/pi) * trap4(i,j)
part2 = part2 - trp2 * av * trap5(i,j)
c
part3 = - factl * vsigf * fact2
part3 = part3 * (m/pi) * trap6(i,j)
part3 = part3 - factl * vsigf * av * trap7(i,j)
c
a(i,j) = part1 + part2 + part3
100 continue
return
c
entry geneb(b)
c
do 101 i = 1, 2
  do 101 j = 1, 2
    b(i,j) = 0.d0
  101 continue

c
av = 2.d0 / height
c
do 200 i = 1, 2
  do 200 j = 3, 4
    if(i.eq.1.and.j.eq.3) then
      b(i,j) = -sigx * av * trap(i,i)
    else
      if(i.eq.1.and.j.eq.4) then
        k = i + 1
        b(i,j) = -sigx * av * trap(i,k)
      else
        if(i.eq.2.and.j.eq.3) then
          b(i,j) = b(i-1,j+1)
        else
          b(i,j) = -sigx * av * trap(i,i)
        endif
      endif
    endif
  200 continue
return
c
entry geneb(c)
c
av = 2.d0 / height
ipl = (mesh * ix(1)) / 100
ip2 = (mesh * ix(2)) / 100
ip3 = (mesh * ix(3)) / 100
c
ipl = ipl + 1
ip2 = ip2 + 1
ip3 = ip3 + 1
c
z1 = ipl * size
z2 = ip2 * size
z3 = ip3 * size
c
tpl = conf * sigb
tp2 = conf * sigb * boron * beta
j = 1
do 102 i = 1, 2
   c(i,j) = sigc(4)*av*pi2(ip3)*dsin(i*pi*z3/height)
   c(i,j+1) = sigc(2)*av*(pi2(ipl)*dsin(i*pi*z1/height) -
                 pi2(ip2)*dsin(i*pi*z2/height))
   c(i,j+2) = -tpl * av * trap9(i)
   c(i,j+3) = tp2*av*trap9(i) - factl*vsigf*av*tral0(i)
do 102
102 continue
return
c
eneny gened(d)
c
av = 2.d0 / height
do 103 i = 1, 2
do 103 j = 1, 4
   if(i.eq.j) then
      d(i,j) = - decai
   else
      d(i,j) = 0.d0
   endif
103 continue
c
i = 3
ii = i - 2
ij = i - 1
j = 1
c
d(i,j) = decai
d(i,j+1) = 0.d0
d(i,j+2) = -decax - sigx*av*trap8(ii,ii)
d(i,j+3) = -sigx*av*trap8(ii,ij)
c
d(i+1,j) = d(i,j+1)
d(i+1,j+1) = d(i,j)
d(i+1,j+2) = d(i,j+3)
d(i+1,j+3) = -dalpha - sigmaav*trapezoid(i,j,i,j)
return

c entry genee(e)

c av = 2.d0 / height
i = 1
ij = i + 1
j = 1

e(i,j) = r1 * sigf
e(i+1,j) = 0.d0
e(i+2,j) = r2 * sigf - sigmaav*trapezoid(i,i)
e(i+3,j) = -sigmaav*trapezoid(i,i)

SUBROUTINE FLUX(Y)
EXTERNAL F
DOUBLE PRECISION ATOL, RWORK, RTOL, T, TOUT, Y, DELZ1, DELZ2, DELB
DOUBLE PRECISION DELT
DIMENSION Y(6), ATOL(6), RWORK(150), IWORK(36)
COMMON/TRANS/DELZ1, DELZ2, DELB, DELT

NEQ = 6
INITIAL VALUE OF THE INDEPENDENT VARIABLE T AND FIRST POINT
WHERE OUTPUT IS DESIRED TOUT

TOUT = 1.0D0
T = 0.d0
ITOL = 2
RTOL = 1.0D-5
DO 20 JJ = 1, NEQ
    ATOL(JJ) = 1.0D-5
20 CONTINUE

OUTPUT AND INPUT OPTION
ITASK = 1
ISTATE = 1
IOPT = 1
IF(IOPT.EQ.1) THEN
DO 21 K = 1, 9
    IWORK(K) = 0
21 CONTINUE

DO 22 K = 1, 20
    RWORK(K) = 0.D0
22 CONTINUE
ENDIF

IWORK(6) = 5000

LENGTH OF RWORK AND IWORK
LRW = 150
LIW = 36

JACOBIAN TYPE INDICATOR JT
JT = 2

CALL LSODA(F,NEQ,Y,T,TOUT,ITOL,RTOL,ATOL,ITASK,ISTATE,ILOPT,
+ RWORK,LRW,IWORK,LIW,JDUM,JT)

IF(ISTATE.LT.0) THEN
    WRITE(9,110) ISTATE
110 FORMAT(IX,'ERROR HALT : ISTATE =',I3)
    STOP
ENDIF

RETURN
END

SUBROUTINE F(NEQ,T,Y,YDOT)
DOUBLE PRECISION T,Y,YD0T,A,B,C,D,E,V,DELZ1,DELZ2,DELB,DELT
DOUBLE PRECISION AY,FACT
DIMENSION Y(6),YD0T(6)
COMMON/TRANS/DELZ1,DELZ2,DELB,DELT
COMMON/TMATIX/A(2,2),B(2,4),C(2,4),D(4,4),E(4,2)
COMMON/TNORM/NTIME
DATA V/7.92D8/

7.92D8 IS THERMAL NEUTRON SPEED 2200*100*3600 CM/HOUR
3.60D3 IS SCALING FACTOR 1 HOUR = 3600 SEC
FACT = NTIME * 3.6D3

YD0T(1)=FACT*V*(A(1,1)*Y(1) + A(1,2)*Y(2) + B(1,1)*Y(3) +
  B(1,2)*Y(4) + B(1,3)*Y(5) + B(1,4)*Y(6) + C(1,1)*DELZ1
  + 2 + C(1,2)*DELZ2 + C(1,3)*DELB + C(1,4)*DELT)
YD0T(2)=FACT*V* ( A(2,1)*Y(1) + A(2,2)*Y(2) + B(2,1)*Y(3) +
\[
\begin{align*}
1B(2,2)*Y(4) + B(2,3)*Y(5) + B(2,4)*Y(6) + C(2,1)*\text{DELZ1} \\
2 + C(2,2)*\text{DELZ2} + C(2,3)*\text{DELB} + C(2,4)*\text{DELT} \\
\text{YD0T}(3) &= \text{FACT}*(D(1,1)*Y(3) + D(1,2)*Y(4) + D(1,3)*Y(5) \\
&\quad + D(1,4)*Y(6) + E(1,1)*Y(1) + E(1,2)*Y(2)) \\
\text{YD0T}(4) &= \text{FACT}*(D(2,1)*Y(3) + D(2,2)*Y(4) + D(2,3)*Y(5) \\
&\quad + D(2,4)*Y(6) + E(2,1)*Y(1) + E(2,2)*Y(2)) \\
\text{YD0T}(5) &= \text{FACT}*(D(3,1)*Y(3) + D(3,2)*Y(4) + D(3,3)*Y(5) \\
&\quad + D(3,4)*Y(6) + E(3,1)*Y(1) + E(3,2)*Y(2)) \\
\text{YD0T}(6) &= \text{FACT}*(D(4,1)*Y(3) + D(4,2)*Y(4) + D(4,3)*Y(5) \\
&\quad + D(4,4)*Y(6) + E(4,1)*Y(1) + E(4,2)*Y(2)) \\
\end{align*}
\]

\begin{verbatim}
DOUBLE PRECISION FUNCTION TRAPP(I,PI1)
DOUBLE PRECISION PI1(I),SIZE,YSUM,Y
COMMON/TRAPE/SIZE,NESH

C WHERE 1.42D14 IS NEUTRON FLUX UNIT
YSUM = 0.DO
DO 300 J = 1, I
  Y = PI1(J) / 1.42D14
  IF(J.EQ.1.OR.J.EQ.I) THEN
    Y = Y / 2.DO
    YSUM = YSUM + Y
  ELSE
    YSUM = YSUM + Y
  ENDIF
300 CONTINUE
TRAPP = YSUM * SIZE
RETURN
END

DOUBLE PRECISION FUNCTION AX0FF(PI2)
DOUBLE PRECISION PI2(61),BEFLUX(31),UPFLUX(31),A01,TRAP
DOUBLE PRECISION SIZE
COMMON/TRAPE/SIZE,NESH

L = (NESH-1)/2
DO 100 I = 1, L
  BEFLUX(I) = PI2(I)
100 CONTINUE

J = 1
DO 101 I = L+1, NESH
  UPFLUX(J) = PI2(I)
  J = J + 1
101 CONTINUE
\end{verbatim}
AO1 = TRAP(L,UPFLUX) - TRAP(L,BEFLUX)
AXOFF = (AO1/TRAP(NESH,P12)) * 1.0D2
RETURN
END

C
DOUBLE PRECISION FUNCTION TRAPX(I,PI1)
DOUBLE PRECISION PI1(I),SIZE,YSUM,Y
COMMON/TRAPE/SIZE,NESH

C
WHERE 2.03D15 IS XENON CONCENTRATION UNIT
YSUM = 0.DO
DO 300 J = 1, I
   Y = PI1(J) / 2.03D15
   IF(J.EQ.1.OR.J.EQ.I) THEN
      Y = Y / 2.DO
      YSUM = YSUM + Y
   ELSE
      YSUM = YSUM + Y
   ENDIF
300 CONTINUE
TRAPX = YSUM * SIZE
RETURN
END

C
DOUBLE PRECISION FUNCTION TRAP(I,PI1)
DOUBLE PRECISION PI1(I),SIZE,YSUM,Y
COMMON/TRAPE/SIZE,NESH

C
YSUM = 0.DO
DO 300 J = 1, I
   Y = PI1(J)
   IF(J.EQ.1.OR.J.EQ.I) THEN
      Y = Y / 2.DO
      YSUM = YSUM + Y
   ELSE
      YSUM = YSUM + Y
   ENDIF
300 CONTINUE
TRAP = YSUM * SIZE
RETURN
END
13. APPENDIX D

13.1 Description for Implementing the XSCD Expert System

Implementation of the XSCD expert system is carried out by passing parameters between the knowledge base and the subsystem of a PWR core control code, PREDICT.FOR, as shown in Figure 1. Core state variables for matching the patterns of power level and shape are transferred from PREDICT.FOR to the knowledge base and control actions for predicting the future behavior of the core state are transferred from the knowledge base to PREDICT.FOR. The interface between PREDICT.FOR and the knowledge base is thus essential to the implementation of the XSCD expert system.

The parameter passing was accomplished using the ASCIIPRM.FOR subroutine, which was linked to the PREDICT.FOR program, with the standard ACTIVATE, SEND, and RETURN commands of INSIGHT2+. The ACTIVATE command activates PREDICT.FOR. The SEND command sends control action parameters to a disk file, PRED.DAT. Then, the PREDICT.FOR program reads the parameters on the disk file PRED.DAT through ASCIIPRM.FOR, predicts the core state data (core state variables) using the parameters, and writes the core state data on the disk file PRED.DAT through the ASCIIPRM.FOR subroutine. The core state data are then received by the knowledge base using the RETURN command.
The execution of PREDICT.FOR requires two inputs; one is the initial full steady state power (100 %) distribution and the other is the scheduled desired power trace as a function of time. The reason PREDICT.FOR requires the initial steady state power distribution is that it solves the linearized time and space dependent diffusion model around the initial steady state operating condition as described in section 5.1. Thus, the procedural requirements before implementing the XSCD expert system are as follows:

1. Installing the initial full steady state power distribution in the disk data base by executing STEADY.FOR.
2. Installing a scheduled desired power trace in the disk data base such as shown in Table 6.

As shown in Figure 1, the graphic engine, incorporated into the XSCD expert system, includes several FORTRAN programs for the display of relative power and xenon distribution, attained power level and AO trace, and adjustments of control rod, boron concentration, and coolant inlet temperature as a function of time. Since the data for
the graphic engine are automatically prepared by executing PREDICT.FOR, each graphic program activated by the ACTIVATE command directly reads the data in the disk data base and then displays the data on the screen of a CRT. That is, the input data for the graphic display are prepared not by the user, but by the system itself.
13.2 Source List of the XSCD Expert System

<table>
<thead>
<tr>
<th>Knowledge Base for Xenon Spacial Control in PWR</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE Xenon Spacial Control Doctor DISPLAY</td>
</tr>
<tr>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>An Expert System for Xenon Spacial Control</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NUMERIC</th>
<th>Time step</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>Initial value control step</td>
</tr>
<tr>
<td>AND</td>
<td>Desired power step</td>
</tr>
<tr>
<td>AND</td>
<td>Power error</td>
</tr>
<tr>
<td>AND</td>
<td>Magnitude of power error</td>
</tr>
<tr>
<td>AND</td>
<td>Power rate</td>
</tr>
<tr>
<td>AND</td>
<td>Desired power</td>
</tr>
<tr>
<td>AND</td>
<td>Current power</td>
</tr>
<tr>
<td>AND</td>
<td>Current xenon</td>
</tr>
<tr>
<td>AND</td>
<td>Previous power</td>
</tr>
<tr>
<td>AND</td>
<td>Desired axial offset</td>
</tr>
<tr>
<td>AND</td>
<td>Current axial offset</td>
</tr>
<tr>
<td>AND</td>
<td>Previous axial offset</td>
</tr>
<tr>
<td>AND</td>
<td>Axial offset error</td>
</tr>
<tr>
<td>AND</td>
<td>Magnitude of axial offset error</td>
</tr>
<tr>
<td>AND</td>
<td>Axial offset rate</td>
</tr>
<tr>
<td>AND</td>
<td>Full length control rod</td>
</tr>
<tr>
<td>AND</td>
<td>Part length control rod</td>
</tr>
<tr>
<td>AND</td>
<td>Boron concentration</td>
</tr>
<tr>
<td>AND</td>
<td>Coolant inlet temperature</td>
</tr>
</tbody>
</table>

| INIT     | Time step = 1 |
| INIT     | Initial value control step = 1 |
| INIT     | Desired power step = 1 |
| INIT     | Initial power = 100 |
| INIT     | Desired power = 1.0 |
| INIT     | Previous power = 1.0 |
| INIT     | Previous axial offset = 1.0 |
| INIT     | Power error = 1.0 |
| INIT     | Power rate = 1.0 |
| INIT     | Magnitude of power error = 1.0 |
| INIT     | Axial offset rate = 1.0 |
Axial offset error = 1.0
Magnitude of axial offset = 1.0
Full length control rod = 0.5
Part length control rod = 0.5
Boron concentration = 0.08
Coolant inlet temperature = 0.18

Control action is taken
And Fortran called
And Outputs displayed

File powerout.txt

1. Control action is taken

Control for done
If Outputs displayed
And Fortran called
And Desired power = 0.0
And Power error = 0.0
And Magnitude of power error = 0.0
And Power rate = 0.0
And Axial offset error = 0.0
And Magnitude of axial offset error = 0.0
And Axial offset rate = 0.0
Then Control action is taken
And Activate POPLT.EXE
And Activate IBMTST.EXE
And Activate A0PLT.EXE
And Activate XEPLT.EXE
And Activate IBMTST.EXE
And Activate RODPLT.EXE
And Activate IBMTST.EXE
And Activate BOPLT.EXE
And Activate IBMTST.EXE
And Activate TEMPLT.EXE
And Activate IBMTST.EXE
And Display Summary
And Stop

For displaying outputs sent by knowledge base
Display Output parameters
Then Outputs displayed
RULE For control beginning 1
IF Outputs displayed
AND Time step = 1
AND Fortran called
AND Power error <= 0.0
AND Desired power <> 100.0
AND Magnitude of power error >= 2.5
AND Axial offset rate = 0.0
AND Axial offset error = 0.0
THEN Control action is taken
AND DISPLAY Satisfied parameters
AND FILE Satisfied parameters
AND Initial value control step := Time step
AND Time step := Time step + 1
AND Previous power := Current power
AND Previous axial offset := Current axial offset
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod
AND Boron concentration := Boron concentration - Be * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For control beginning 2
IF Outputs displayed
AND Time step = 1
AND Fortran called
AND Desired power <> 100.0
AND Power error <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate = 0.0
AND Axial offset error = 0.0
THEN Control action is taken
AND DISPLAY Satisfied parameters
AND FILE Satisfied parameters
AND Initial value control step := Time step
AND Time step := Time step + 1
AND Previous power := Current power
AND Previous axial offset := Current axial offset
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod
AND Boron concentration := Boron concentration - Be * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For control beginning 3
IF Outputs displayed
AND Time step = 1
AND Fortran called
AND Desired power <> 100.0
AND Power error <= 0.0
AND Magnitude of power error < 1.0
AND Axial offset rate = 0.0
AND Axial offset error = 0.0
THEN Control action is taken
AND DISPLAY Satisfied parameters
AND FILE Satisfied parameters
AND Initial value control step := Time step
AND Time step := Time step + 1
AND Previous power := Current power
AND Previous axial offset := Current axial offset
AND Full length control rod := Full length control rod - FLcr
AND Part length control rod := Part length control rod
AND Boron concentration := Boron concentration - BC
AND Coolant inlet temperature := Coolant inlet temperature + CIT
AND CYCLE

RULE For control beginning 4
IF Outputs displayed
AND Time step = 1
AND Fortran called
AND Desired power = 100.0
AND Power error <= 0.0
AND Magnitude of power error < 1.0
AND Axial offset rate = 0.0
AND Axial offset error = 0.0
THEN Control action is taken
AND DISPLAY Satisfied parameters
AND FILE Satisfied parameters
AND Initial value control step := Time step
AND Time step := Time step + 1
AND Previous power := Current power
AND Previous axial offset := Current axial offset
AND Full length control rod := 0.0
AND Part length control rod := 0.0
AND Boron concentration := 0.0
AND Coolant inlet temperature := 0.0
AND CYCLE

RULE For calling FORTRAN sub-system
ACTIVATE PREDICT.EXE
DISK PRED.DAT
SEND Time step
SEND Initial value control step
SEND Desired power step
SEND Previous power
SEND Previous axial offset
SEND Full length control rod
SEND Part length control rod
SEND Boron concentration
SEND Coolant inlet temperature
RETURN Time step
RETURN Desired power step
RETURN Desired power
RETURN Current power
RETURN Current xenon
RETURN Desired axial offset
RETURN Current axial offset
RETURN Power error
RETURN Magnitude of power error
RETURN Power rate
RETURN Axial offset error
RETURN Magnitude of axial offset error
RETURN Axial offset rate
THEN Fortran called

RULE For satisfied power and axial offset control Al-1
IF Outputs displayed
AND Fortran called
AND Magnitude of power error < 1.0
AND Magnitude of axial offset error < 0.5
AND Desired power <> 100.0
THEN Control action is taken
AND DISPLAY Satisfied parameters
AND FILE Satisfied parameters
AND ACTIVATE NEUDATA.EXE ACTIVATE IBMTST.EXE
AND ACTIVATE XENDATA.EXE
AND ACTIVATE IBMTST.EXE
AND Initial value control step := Time step
AND Time step := Time step + 1
AND Previous power := Current power
AND Previous axial offset := Current axial offset
AND Desired power step := Desired power step + 1
AND Initial power := Desired power
AND CYCLE

RULE For satisfied power and axial offset control Al-2
IF Outputs displayed
AND Fortran called
AND Magnitude of power error < 1.0
AND Magnitude of axial offset error < 0.5
AND Desired power = 100.0
THEN Control action is taken
AND DISPLAY Satisfied parameters
AND FILE Satisfied parameters
AND ACTIVATE NEUDATA.EXE
AND ACTIVATE IBMTST.EXE
AND ACTIVATE XENDATA.EXE
AND ACTIVATE IBMTST.EXE
AND Initial value control step := Time step
AND Time step := Time step + 1
AND Previous power := Current power
AND Previous axial offset := Current axial offset
AND Desired power step := Desired power step + 1
AND Initial power := Desired power
AND Full length control rod := 0.0
AND Part length control rod := 0.0
AND Boron concentration := 0.0
AND Coolant inlet temperature := 0.0
AND CYCLE

RULE For power and axial offset pattern learning 311
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Plcr * 12
AND Part length control rod := Part length control rod + Plcr * 16
AND Boron concentration := Boron concentration - Be * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 312
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration - Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 313
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod + Flcr
AND Boron concentration := Boron concentration - Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 321
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod + Flcr * 16
AND Boron concentration := Boron concentration - Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 322
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration - Be * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 323
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod + Plcr
AND Boron concentration := Boron concentration - Be * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 331
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod + Plcr * 16
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE
RULE For power and axial offset pattern learning 332
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 3311
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod + Plcr * 12
AND Boron concentration := Boron concentration - Be * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 3312
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration - Be * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 3313
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod + Plcr
AND Boron concentration := Boron concentration - Be * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 3321
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod + Plcr * 12
AND Boron concentration := Boron concentration - Be * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 3322
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration - Be * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 3323
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod + Plcr
AND Boron concentration := Boron concentration - Be * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 3331
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod + Plcr * 12
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 3332
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 33311
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod - Flcr * 12
AND Boron concentration := Boron concentration - Be * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 33312
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration - Be * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE
RULE For power and axial offset pattern learning 33313
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod - Flcr
AND Boron concentration := Boron concentration - Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 33321
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod - Flcr * 12
AND Boron concentration := Boron concentration - Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 33322
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration - Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 33323
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod - Flcr
AND Boron concentration := Boron concentration - Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 33331
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod - Flcr * 12
AND Boron concentration := Boron concentration - Bc
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 33332
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration - Bc
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 333311
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod - Flcr * 16
AND Boron concentration := Boron concentration - Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 333312
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration - Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 333313
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 12
AND Part length control rod := Part length control rod - Flcr
AND Boron concentration := Boron concentration - Be * 4
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 333321.
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod - Flcr * 16
AND Boron concentration := Boron concentration - Be * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 333322
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length control rod := Part length control rod
AND Boron concentration := Boron concentration - Be * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE
RULE For power and axial offset pattern learning 333323
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 4
AND Part length rod := Part length control rod - Plcr
AND Boron concentration := Boron concentration - Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 333331
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod - Plcr
AND Boron concentration := Boron concentration - Bc
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 333332
IF Outputs displayed
AND Fortran called
AND Power error <= 0.0
AND Power rate <= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 411
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod + Flcr * 16
AND Boron concentration := Boron concentration - Be * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 412
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration - Be * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 413
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod + Plcr
AND Boron concentration := Boron concentration - Bc * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 421
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod + Plcr * 16
AND Boron concentration := Boron concentration - Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 422
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration - Be * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 423
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod + Plcr
AND Boron concentration := Boron concentration - Be * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 431
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod + Plcr * 16
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 432
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 4411
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod + Flcr * 12
AND Boron concentration := Boron concentration - Be * 6
AND Coolant inlet temperature := Coolant temperature + Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 4412
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration - Be * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE
RULE For power and axial offset pattern learning 4413
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod + Flcr
AND Boron concentration := Boron concentration - Bc * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE
RULE For power and axial offset pattern learning 4421
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >=0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod + Flcr
AND Boron concentration := Boron concentration - Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE
RULE For power and axial offset pattern learning 4422
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration - Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 4423
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod + Plcr
AND Boron concentration := Boron concentration - Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 4431
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod + Plcr * 12
AND Boron concentration := Boron concentration - Bc
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 4432
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 44411
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod - Plcr * 12
AND Boron concentration := Boron concentration - Be * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 44412
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod - Plcr * 4
AND Boron concentration := Boron concentration - Be * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE
RULE For power and axial offset pattern learning 44413
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod - Plcr
AND Boron concentration := Boron concentration - Bc * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 44421
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod - Plcr * 12
AND Boron concentration := Boron concentration - Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 44422
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND CYCLE
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod - Plcr * 4
AND Boron concentration := Boron concentration - Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 44423
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod - Plcr
AND Boron concentration := Boron concentration - Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 44431
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod - Plcr * 12
AND Boron concentration := Boron concentration - Bc
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 44432
IF Outputs displayed 
AND Fortran called 
AND Power error < 0.0 
AND Power rate > 0.0 
AND Magnitude of power error < 1.0 
AND Magnitude of power error >= 0.0 
AND Axial offset rate < 0.0 
AND Axial offset error > 0.0 
AND Magnitude of axial offset error < 1.0 
AND Magnitude of axial offset error >= 0.5 
THEN Control action is taken 
AND Initial value control step := Time step 
AND Full length control rod := Full length control rod - Flcr 
AND Part length control rod := Part length control rod - Flcr * 4 
AND Boron concentration := Boron concentration - Be 
AND Coolant inlet temperature := Coolant inlet temperature + Cit 
AND CYCLE

RULE For power and axial offset pattern learning 444411 
IF Outputs displayed 
AND Fortran called 
AND Power error < 0.0 
AND Power rate > 0.0 
AND Magnitude of power error >= 2.5 
AND Axial offset rate > 0.0 
AND Axial offset error > 0.0 
AND Magnitude of axial offset error >= 1.0 
THEN Control action is taken 
AND Initial value control step := Time step 
AND Full length control rod := Full length control rod - Flcr * 16 
AND Part length control rod := Part length control rod - Flcr * 16 
AND Boron concentration := Boron concentration - Be * 6 
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12 
AND CYCLE

RULE For power and axial offset pattern learning 444412 
IF Outputs displayed 
AND Fortran called 
AND Power error < 0.0 
AND Power rate > 0.0 
AND Magnitude of power error >= 2.5 
AND Axial offset rate > 0.0 
AND Axial offset error > 0.0 
AND Magnitude of axial offset error < 1.0 
AND Magnitude of axial offset error >= 0.5 
THEN Control action is taken 
AND Initial value control step := Time step 
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod - Plcr * 4
AND Boron concentration := Boron concentration - Bc * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 444413
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 16
AND Part length control rod := Part length control rod - Plcr
AND Boron concentration := Boron concentration - Bc * 6
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 444421
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod - Plcr * 16
AND Boron concentration := Boron concentration - Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 444422
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod - Plcr * 4
AND Boron concentration := Boron concentration - Be * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 444423
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr * 5
AND Part length control rod := Part length control rod - Plcr
AND Boron concentration := Boron concentration - Be * 3
AND Coolant inlet temperature := Coolant inlet temperature + Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 444431
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod - Plcr * 16
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 444432
IF Outputs displayed
AND Fortran called
AND Power error < 0.0
AND Power rate > 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod - Flcr
AND Part length control rod := Part length control rod - Plcr * 4
AND Boron concentration := Boron concentration - Be
AND Coolant inlet temperature := Coolant inlet temperature + Cit
AND CYCLE

RULE For power and axial offset pattern learning 511
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod + Plcr * 16
AND Boron concentration := Boron concentration + Be * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 512
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error $\geq 0.5$
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration + Bc * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 513
IF Outputs displayed
AND Fortran called
AND Power error $> 0.0$
AND Power rate $< 0.0$
AND Magnitude of power error $\geq 2.5$
AND Axial offset rate $< 0.0$
AND Axial offset error $< 0.0$
AND Magnitude of axial offset error $< 0.5$
AND Magnitude of axial offset error $\geq 0.0$
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod + Flcr
AND Boron concentration := Boron concentration + Bc * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 521
IF Outputs displayed
AND Fortran called
AND Power error $> 0.0$
AND Power rate $< 0.0$
AND Magnitude of power error $< 2.5$
AND Magnitude of power error $\geq 1.0$
AND Axial offset rate $< 0.0$
AND Axial offset error $< 0.0$
AND Magnitude of axial offset error $\geq 1.0$
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
AND Part length control rod := Part length control rod + Flcr * 16
AND Boron concentration := Boron concentration + Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 522
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration + Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 523
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration + Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 531
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod + Flcr * 16
AND Boron concentration := Boron concentration + Bc
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 532
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration + Bc
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 5511
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod + Flcr * 12
AND Boron concentration := Boron concentration + Bc * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 5512
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration + Be * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 5513
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error > 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod + Flcr
AND Boron concentration := Boron concentration + Be * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 5521
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error > 1.0
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
AND Part length control rod := Part length control rod + Flcr * 12
AND Boron concentration := Boron concentration + Be * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE
RULE For power and axial offset pattern learning 5522
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration + Br * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 5523
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
AND Part length control rod := Part length control rod + Flcr
AND Boron concentration := Boron concentration + Br * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 5531
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod + Flcr * 12
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 5532
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 55511
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod - Flcr * 12
AND Boron concentration := Boron concentration + Be * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 55512
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration + Be * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 55513
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration + Be * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 55521
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
RULE For power and axial offset pattern learning 55522
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Plcr * 5
AND Part length control rod := Part length control rod - Plcr * 4
AND Boron concentration := Boron concentration + Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 55523
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Plcr * 5
AND Part length control rod := Part length control rod - Plcr
AND Boron concentration := Boron concentration + Bc * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 55531
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod - Plcr * 12
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 55532
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod - Plcr * 4
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 555511
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod - Plcr * 16
AND Boron concentration := Boron concentration + Be * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 555512
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration + Be * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 555513
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 16
AND Part length control rod := Part length control rod - Flcr
AND Boron concentration := Boron concentration + Be * 6
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 12
AND CYCLE

RULE For power and axial offset pattern learning 555521
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
AND Part length control rod := Part length control rod - Flcr * 16
AND Boron concentration := Boron concentration + Be * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 555522
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration + Be * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE

RULE For power and axial offset pattern learning 555523
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 5
AND Part length control rod := Part length control rod - Flcr
AND Boron concentration := Boron concentration + Be * 3
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 3
AND CYCLE
RULE For power and axial offset pattern learning 555531
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod - Plcr * 16
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 555532
IF Outputs displayed
AND Fortran called
AND Power error > 0.0
AND Power rate < 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod - Plcr * 4
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 611
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + F1cr * 12
AND Part length control rod := Part length control rod + Plcr * 16
AND Boron concentration := Boron concentration + Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 612
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + F1cr * 12
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration + Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 613
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + F1cr * 12
AND Part length control rod := Part length control rod + Plcr
AND Boron concentration := Boron concentration + Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 621
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod + Plcr * 16
AND Boron concentration := Boron concentration + Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 622
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration + Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 623
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod + Plcr
AND Boron concentration := Boron concentration + Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 631
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod + Plcr * 16
AND Boron concentration := Boron concentration + Bc
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 632
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error < 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod + Plcr * 4
AND Boron concentration := Boron concentration + Bc
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 6611
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 12
AND Part length control rod := Part length control rod + Plcr * 12
AND Boron concentration := Boron concentration + Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 6612
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 12
AND Part length control rod := Part length control rod + Plcr * 12
AND Boron concentration := Boron concentration + Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 6613
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 12
AND Part length control rod := Part length control rod + Plcr
AND Boron concentration := Boron concentration + Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE
RULE For power and axial offset pattern learning 6621
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod + Plcr * 12
AND Boron concentration := Boron concentration + Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 6622
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Controls taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod + Plcr * 12
AND Boron concentration := Boron concentration + Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 6623
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod + Flcr
AND Boron concentration := Boron concentration + Be * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 6631
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod + Flcr * 12
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 6632
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate >= 0.0
AND Axial offset error <= 0.0
AND Magnitude of axial offset error <= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod + Flcr * 4
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 66611
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 12
AND Part length control rod := Part length control rod - Flcr * 12
AND Boron concentration := Boron concentration + Be * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 66612
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 12
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration + Be * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 66613
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 12
AND Part length control rod := Part length control rod - Flcr * 12
AND Boron concentration := Boron concentration + Be * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 66621
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod - Plcr * 12
AND Boron concentration := Boron concentration + Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 66622
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod - Plcr * 12
AND Boron concentration := Boron concentration + Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 66623
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod - P1cr
AND Boron concentration := Boron concentration + Be * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 66631
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod - P1cr * 12
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 66632
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate < 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error <= 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod - P1cr * 4
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 666611
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 12
AND Part length control rod := Part length control rod - Flcr * 16
AND Boron concentration := Boron concentration + Be * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 666612
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 12
AND Part length control rod := Part length control rod - Flcr * 16
AND Boron concentration := Boron concentration + Be * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 666613
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error >= 2.5
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 12
AND Part length control rod := Part length control rod - Flcr
AND Boron concentration := Boron concentration + Bc * 4
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 10
AND CYCLE

RULE For power and axial offset pattern learning 666621
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod - Flcr * 16
AND Boron concentration := Boron concentration + Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 666622
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration + Bc * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 666623
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 2.5
AND Magnitude of power error >= 1.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 0.5
AND Magnitude of axial offset error >= 0.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr * 4
AND Part length control rod := Part length control rod - Plcr
AND Boron concentration := Boron concentration + Be * 2
AND Coolant inlet temperature := Coolant inlet temperature - Cit * 2
AND CYCLE

RULE For power and axial offset pattern learning 666631
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error >= 1.0
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod - Plcr * 16
AND Boron concentration := Boron concentration + Be
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

RULE For power and axial offset pattern learning 666632
IF Outputs displayed
AND Fortran called
AND Power error >= 0.0
AND Power rate >= 0.0
AND Magnitude of power error < 1.0
AND Magnitude of power error >= 0.0
AND Axial offset rate > 0.0
AND Axial offset error > 0.0
AND Magnitude of axial offset error < 1.0
AND Magnitude of axial offset error >= 0.5
THEN Control action is taken
AND Initial value control step := Time step
AND Full length control rod := Full length control rod + Flcr
AND Part length control rod := Part length control rod - Flcr * 4
AND Boron concentration := Boron concentration + Bc
AND Coolant inlet temperature := Coolant inlet temperature - Cit
AND CYCLE

DISPLAY Satisfied parameters
Satisfied power, axial offset and control values
Time = [Time step]
Desired power = [Desired power]
Current power = [Current power]
Current xenon = [Current xenon]
Desired AO = [Desired axial offset]
Current AO = [Current axial offset]
Full length control rod = [Full length control rod]
Part length control rod = [Part length control rod]
Boron concentration = [Boron concentration]
Coolant inlet temperature = [Coolant inlet temperature]

DISPLAY Summary
Control action is finished
Optimal control strategy for the desired power trace is at POWEROUT.TXT

DISPLAY Output parameters
Control parameters decided by knowledge base
Time step = [Time step]
Desired power step = [Desired power step]
Previous power = [Previous power]
Previous axial offset = [Previous axial offset]
Full length control rod = [Full length control rod]
Part length control = [Part length control rod]
Boron concentration = [Boron concentration]
Coolant inlet temperature = [Coolant inlet temperature]
14. APPENDIX E. SOURCE LIST OF GRAPHIC PROGRAMS

PROGRAM TO PLOT THE REACTOR POWER

DIMENSION POWER1(40),POWER2(40),X(40),NODE(40)
CHARACTER ANS
CHARACTER*40 DATNAM
DATA DATNAM/'SIMPLT.DAT'/

ANS = S

OPEN(14,FILE=DATNAM,STATUS='NEW',FORM='UNFORMATTED')

IF(ANS.EQ.'S'.OR.ANS.EQ.'s') CALL SCREEN(XSIZE,YSIZE)
IF(ANS.EQ.'P'.OR.ANS.EQ.'p') CALL PRINTR(XSIZE,YSIZE)
IF(ANS.NE.'S'.AND.ANS.NE.'P') CALL SCREEN(XSIZE,YSIZE)

C READING THE INPUT DATA FROM FILE OF POAO.OUT

OPEN(1,FILE='POAO.OUT',STATUS='OLD',ACCESS='DIRECT', + FORM='FORMATTED',RECL=55)

I = 1
88 READ(1,600,REC=I,END=500) NODE(I),POWER1(I),POWER2(I)
X(I) = FLOAT(NODE(I))
I = I + 1
GO TO 88

500 NPTS = I - 1
600 FORMAT(I2,1X,2(F8.3,1X))

CALL GRAPH(NPTS,X,POWER1,' ',2,XSIZE,YSIZE,0.0,0.0,10.0,60.0, + 'TIME(HOUR);','REACTOR POWER(%);',';',';')
CALL GRAPHS(NPTS,X,POWER2,2,101,'CURRENT POWER;')

STOP
END

PROGRAM TO PLOT THE AXIAL OFFSET

DIMENSION A01(40),A02(40),X(40),NODE(40)
CHARACTER ANS
CHARACTER*40 DATNAM
DATA DATNAM/'SIMPLT.DAT'/

ANS = S
OPEN(14,FILE=DATNAM,STATUS='NEW',FORM='UNFORMATTED')

IF(ANS.EQ.'S'.OR.ANS.EQ.'S') CALL SCREEN(XSIZE,YSIZE)
IF(ANS.EQ.'P'.OR.ANS.EQ.'p') CALL PRINTR(XSIZE,YSIZE)
IF(ANS.NE.'S'.AND.ANS.NE.'P') CALL SCREEN(XSIZE,YSIZE)

C READING THE INPUT DATA FROM FILE OF POAO.OUT

OPEN(1,FILE='POAO.OUT',STATUS='OLD',ACCESS='DIRECT',
+ FORM='FORMATTED',RECL=55)

I = 1
88 READ(1,600,REC=I,END=500) NODE(I),A01(I),A02(I)
     X(I) = FLOAT(NODE(I))
     I = I + 1
GO TO 88

500 NPTS = I - 1
600 FORMAT(I2,28X,2(F8.3,1X))

CALL GRAPH(NPTS,X,A01,' ',2,XSIZE,YSIZE,0.0,0.0,2.0,-16.0,
+ 'TIME(HOUR)';', 'AXIAL OFFSET(%);'';';';')
CALL GRAPHS(NPTS,X,A02,2,101,'CURRENT AO;')

STOP
END

C PROGRAM TO PLOT THE CONTROL ROD MOTION

DIMENSION ROD1(40),ROD2(40),X(40),NODE(40)
CHARACTER ANS
CHARACTER*40 DATNAM
DATA DATNAM/'SIMPLT.DAT'/

ANS = S

OPEN(14,FILE=DATNAM,STATUS='NEW',FORM='UNFORMATTED')

IF(ANS.EQ.'S'.OR.ANS.EQ.'S') CALL SCREEN(XSIZE,YSIZE)
IF(ANS.EQ.'P'.OR.ANS.EQ.'p') CALL PRINTR(XSIZE,YSIZE)
IF(ANS.NE.'S'.AND.ANS.NE.'P') CALL SCREEN(XSIZE,YSIZE)

C READING THE INPUT DATA FROM FILE OF CONTROL.OUT

OPEN(2,FILE='CONTROL.OUT',STATUS='OLD',ACCESS='DIRECT',
+ FORM='FORMATTED',RECL=45)
PROGRAM TO PLOT THE BORON CONCENTRATION

DIMENSION BORON1(40),X(40),NODE(40)
CHARACTER*40 DATNAM
DATA DATNAM/'SIMPLT.DAT'/

ANS = S
OPEN(14,FILE=DATNAM,STATUS='NEW',FORM='UNFORMATTED')

IF(ANS.EQ.'S'.OR.ANS.EQ.'s') CALL SCREEN(XSIZE,YSIZE)
IF(ANS.EQ.'P'.OR.ANS.EQ.'p') CALL PRINTR(XSIZE,YSIZE)
IF(ANS.NE.'S'.AND.ANS.NE.'P') CALL SCREEN(XSIZE,YSIZE)

READING THE INPUT DATA FROM FILE OF POAO.OUT
OPEN(2,FILE='CONTROL.OUT',STATUS='OLD',ACCESS='DIRECT',
     FORM='FORMATTED',RECL=45)

I = 1
88 READ(2,600,REC=I,END=500) NODE(I),BORON1(I)
   X(I) = FLOAT(NODE(I))
   I = I + 1
   GO TO 88

500 NPTS = I - 1
600 FORMAT(I2,1X,2(F8.3,1X))

CALL GRAPH(NPTS,X,BORON1,';',2,XSIZE,YSIZE,0.0,0.0,25.0,-100.0,
   + 'TIME(HOUR);','BORON(PPM);','BORON CONCENTRATION;',';')
STOP
END

PROGRAM TO PLOT THE COOLANT INLET TEMP

DIMENSION CTEMP1(40),X(40),NODE(40)
CHARACTER ANS
CHARACTER*40 DATNAM
DATA DATNAM/'SIMPLT.DAT'/

ANS = S

OPEN(14,FILE=DATNAM,STATUS='NEW',FORM='UNFORMATTED')

IF(ANS.EQ.'S'.OR.ANS.EQ.'s') CALL SCREEN(XSIZE,YSIZE)
IF(ANS.EQ.'P'.OR.ANS.EQ.'p') CALL PRINTR(XSIZE,YSIZE)
IF(ANS.NE.'S'.AND.ANS.NE.'P') CALL SCREEN(XSIZE,YSIZE)

READING THE INPUT DATA FROM FILE OF POAO.OUT

OPEN(2,FILE='CONTROL.OUT',STATUS='OLD',ACCESS='DIRECT',
     FORM='FORMATTED',RECL=45)

I = 1
88 READ(2,600,REC=I,END=500) NODE(I),CTEMP1(I)
   X(I) = FLOAT(NODE(I))
   I = I + 1
   GO TO 88

500 NPTS = I - 1
600 FORMAT(I2,28X,F8.3)

CALL GRAPH(NPTS,X,CTEMP1,';','2',XSIZE,YSIZE,0.0,0.0,5.0,-5.0,
     'TIME(HOUR);','TEMPERATURE(C);','COOLANT TEMPERATURE;';')

STOP
END

program to plot the relative neutron flux distribution in the z direction

DIMENSION FLUX1(61),FLUX2(61),NODE(61),X(61)
CHARACTER ANS
CHARACTER*40 DATNAM
DATA DATNAM/'SIMPLT.DAT'/
ANS = S

OPEN(14,FILE=DATNAM,STATUS='NEW',FORM='UNFORMATTED')

IF(ANS.EQ.'S'.OR.ANS.EQ.'s') CALL SCREEN(XSIZE,YSIZE)
IF(ANS.EQ.'P'.OR.ANS.EQ.'p') CALL PRINTR(XSIZE,YSIZE)
IF(ANS.NE.'S'.AND.ANS.NE.'P') CALL SCREEN(XSIZE,YSIZE)

reading the steady and transient state data
1.42d14 is a unit of neutron flux used in the model.

OPEN(8,FILE='TRANSI.OUT',STATUS='OLD')
OPEN(9,FILE='STEADY.DAT',STATUS='OLD')

READ(8,*) MESH
DO 89 I = 1, MESH
      READ(8,*) NODE(I),FLUX1(I)
      READ(9,98) FLUX2(I)
      X(I) = FLOAT(NODE(I))
      FLUX2(I) = FLUX2(I) / 1.42D14
89 CONTINUE
98 FORMAT(E11.4)

CALL GRAPH(MESH,X,FLUX2,2,102,XSIZE,YSIZE,0.0,0.0,0.0,0.0,0.0,
+ 'AXIAL DIRECTION;','RELATIVE POWER;',';','STEADY;')
CALL GRAPHS(MESH,X,FLUX1,4,2,')

STOP
END

program to plot the relative xenon distribution in the z direction

DIMENSION FLUXI(61),FLUX2(61),NODE(61),X(61)
CHARACTER ANS
CHARACTER*40 DATNAM
DATA DATNAM/'SIMPLT.DAT'/

ANS = S

OPEN(14,FILE=DATNAM,STATUS='NEW',FORM='UNFORMATTED')

IF(ANS.EQ.'S'.OR.ANS.EQ.'s') CALL SCREEN(XSIZE,YSIZE)
IF(ANS.EQ.'P'.OR.ANS.EQ.'p') CALL PRINTR(XSIZE,YSIZE)
IF(ANS.NE.'S'.AND.ANS.NE.'P') CALL SCREEN(XSIZE,YSIZE)
reading the steady and transient state data
2.03D15 is a unit of XENON DENSITY used in the model.

OPEN(8,FILE='TRANS2.OUT',STATUS='OLD')
OPEN(9,FILE='STEADY.DAT',STATUS='OLD')

READ(8,*) MESH
DO 89 I = 1, MESH
    READ(8,*) NODE(I),FLUX1(I)
    READ(9,98) FLUX2(I)
    X(I) = FLOAT(NODE(I))
    FLUX2(I) = FLUX2(I) / 2.03D15
CONTINUE
98 FORMAT(13X,E11.4)

CALL GRAPH(MESH,X,FLUX2,2,102,XSIZE,YSIZE,0.0,0.0,0.0,0.0,0.0, + 'AXIAL DIRECTION;', 'RELATIVE XENON;', ';', 'STEADY;')
CALL GRAPHS(MESH,X,FLUX1,4,2,';')

STOP
END

PROGRAM TO PLOT THE XENON DENSITY

DIMENSION XENON1(40),X(40),NODE(40)
CHARACTER ANS
CHARACTER*40 DATNAM
DATA DATNAM/'SIMPLICIT.DAT'/

ANS = S
OPEN(14,FILE=DATNAM,STATUS='NEW',FORM='UNFORMATTED')

IF(ANS.EQ.'S'.OR.ANS.EQ.'s') CALL SCREEN(XSIZE,YSIZE)
IF(ANS.EQ.'P'.OR.ANS.EQ.'p') CALL PRINTR(XSIZE,YSIZE)
IF(ANS.NE.'S'.AND.ANS.NE.'P') CALL SCREEN(XSIZE,YSIZE)

READING THE INPUT DATA FROM FILE OF POAO.OUT

OPEN(1,FILE='POAO.OUT',STATUS='OLD',ACCESS='DIRECT', + FORM='FORMATTED',RECL=55)

I = 1
88 READ(1,600,REC=I,END=500) NODE(I),XENON1(I)
    X(I) = FLOAT(NODE(I))
    I = I + 1
GO TO 88

500 NPTS = I - 1
600 FORMAT(I2,19X,F8.3)

CALL GRAPH(NPTS,X,XENON1,';',2,XSIZE,YSIZE,0.0,0.0,5.0,85.0,
+ 'TIME(HOUR);','XENON(%);','INTEGRATED XENON;'',';')

STOP
END