AN ABSTRACT OF THE THESIS OF

Kevin J. Greek for the degree of Doctor of Philosophy in Nuclear Engineering presented on December 12, 1989.

Title: Development of an Object-Oriented Expert System for PWR Core Reload

Abstract approved: Redacted for privacy

Dr. Alan Robinson

Reactor refueling is a computationally intensive problem solving process that requires automation since significant resources are expended upon the search for an optimal core loading. Without imposing some constraints upon the configurations investigated, an extremely large number of prototype loadings may be generated without discovery of an improved loading. As the result of years of study upon the problem, a fuel management expert understands how to direct and constrain the search for an acceptable minimum peak power loading. This research attempts to automate the expert's knowledge with the ease of representation and maintenance the tools of artificial intelligence (AI) are most specialized. It seeks to make his expertise generally available.

The structure, operators, and search methods for representation of the core reload problem are identified and reveal the limitations which many expert system tools have for its solution. An object-oriented

representation allows a natural means to define components of the problem and share many dependent attributes of objects in the representation consistently. The expert system prototype, Shuffle, is written in Smalltalk, an object-oriented programming language, and evaluates loadings as it generates them using a two group, two dimensional power calculation compiled in a PC-based FORTRAN.

Proven strategies of fuel management experts were incorporated on top of an object oriented representation in a highly interactive environment. Shuffle currently includes three strategies or subgoals, each subgoal progressively positioning less reactive fuel by distinct move instructions. Each subgoal is implemented as a hierarchical subclass of constraints and heuristics that generally remain consistent with each new loading. These constraints include requirements for a modified out-in loading pattern with map regions declared as even, odd, intermediate, and peripheral.

Evaluating the intelligence of Shuffle requires an analysis of both its rate of convergence toward an improved pattern and its ability to correct what appears to be a poor pattern. Some experiments with test patterns revealed that additional constraints could improve convergence but may limit exploration by the system.

Development of an Object-Oriented Expert System for PWR Core Reload

by

Kevin J. Greek

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Completed December 12, 1989

Commencement June 1990

Approved:

Redacted for privacy

Professor of Nuclear Engineering in charge of major

Redacted for privacy

Head of Department of Nuclear Engineering

Redacted for privacy

Dean of Graduate School

Date thesis is presented December 12, 1989

Typed by Kevin Greek for Kevin J. Greek

ACKNOWLEDGMENT

The culmination of this research and my education cannot be without the support and encouragement of so many. It is impossible to adequately thank everyone in this short space so I will just briefly mention a few. I first express my appreciation to my major professor Dr. Alan H. Robinson for the monetary and moral support during the time required to extend myself from an electrical engineering technician to nuclear engineer. On several occasions his advice and assistance were invaluable. I wish to recognize Byung-Oh Cho for his instrumental assistance toward preparation of the ADI power calculation code used in this research. Also great thanks to Joan Heaberlin who contributed many of the technical articles needed to support this research.

I dedicate this work to my parents, whose self-sacrifice and commitment always extended beyond our family. No dedication could be complete without mention of my grandmother whose many years of patience, unselfishness, and love set a precedent for me that I have sought to follow. Finally, I wish to dedicate this work to the many wonderful international students with who I have made friends here, who enlightened me with their experiences and nation's customs, who sought my help, and who left before me. It was here staying the extra time to share in their endeavors where I found my greatest reward.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

DEVELOPMENT OF AN OBJECT-ORIENTED EXPERT SYSTEM FOR PWR CORE RELOAD

I. INTRODUCTION

Nuclear fuel management has been defined as the collection of practices and principles necessary for the planning, scheduling, refueling, and safe operations of nuclear power plants while seeking to reduce total plant and system costs through the timely procurement of nuclear fuel and related services. It can be divided into two major subdivisions: out-of-core management and in-core management. The major efforts of out-of-core fuel management are contracting and purchasing services such as conversion, enrichment, fabrication, and spent fuel disposal. On the other hand, in-core fuel management attempts to optimize nuclear fuel utilization within the reactor core so as to meet the required licensing and operational constraints and still maintain an economic advantage. In essence, successful in-core nuclear fuel management requires an extensive knowledge of economics, neutronics, materials, and plant availability. Its planning must possess a margin for adaptability should expectations change.

Nuclear fuel management is undergoing a change of emphasis as utilities seek to maximize fuel use and power plant availability. Besides economic considerations, this attitude is consistent with the

trend of national policy, which is opposed to reprocessing spent fuel. As long as fuel reprocessing was expected to take place, there was little reason for concern about fuel utilization in commercial reactors, since reprocessing would have permitted recovery of uranium for recycle at an enrichment plant and recovery of plutonium for use as fuel. With the decision to store spent fuel indefinitely, improved fuel utilization assumes the utmost importance. Annual refueling has been the practice in this country. At each refueling, approximately 1/3 of all bundles in a pressurized water reactor (PWR) or 1/4 of all bundles in a boiling water reactor (BWR) core are replaced and the fuel assemblies are shuffled to better locations to increase overall burnup. But, in the interest of improved utilization, there is a need to extend useful operation up to 18 months.

Due to the complexity of economic and engineering requirements, complete optimization of a refueling scheme for even a single cycle is a difficult task. First, the core reload design must meet the utility's specification of energy for a maximized cycle lifetime. Although the procurement of materials and manpower resources is also an important aspect of in-core fuel management, the major design parameters that optimize the reload design include the loading pattern and the

in-core power balance it imposes, the range of enrichments among fuel assemblies available for selection, the number and position of burnable absorbers for control of reactivity and power peaking, and the inventory of depleted fuel. In addition, there are four safety criteria that must be satisfied by the reload design. These are: safe control margins during shutdown; ejected rod worth; design burn up limits; and fuel performance characteristics. A reload design that satisfies these diverse and almost contradictory constraints is unlikely to take the form of a simple formula or code.

When a core reload design is required for any given cycle, an in-core fuel management expert typically must combine his judgement with available tools to scope, evaluate, and refine the reload plan. The expert performs an iterative task when refining a proposed loading to meet constraints, sometimes modifying his strategy according to new requirements. Although several programs now in standard practice evaluate a proposed core loading for neutronic and thermal-hydraulic behavior, scoping tools are needed to determine plans that meet all circumstances more comprehensively. The expert's task could be simplified if his reasoning process could be automated. Overall, a planning program for the core reload problem is needed.

It must be versatile and flexible enough to satisfy the constraints for plant operation under actual circumstances.

This paper presents the development of a PWR core reload expert system in which the rules the fuel mamagement expert uses to direct the search to an optimal loading pattern has been encoded. The expert system uses an interactive graphics environment and has a limited explanation capability. Its object-oriented representation enables simplified modification of constraints to meet changing reactor specifications or refueling strategies.

II. BACKGROUND

II.A Loading behavior

A characteristic parameter of nuclear fission reactions used to define criticality is the multiplication factor, k, which is taken as the ratio of the number of neutrons in one generation to the number of neutrons in the previous generation. Reactor configuration, the mass of fissile fuel, and the number of neutrons that are not lost but contribute to fission, all influence the multiplication factor. Two characterizations of multiplication factors exist: the infinite multiplication factor, k-infinity, which assumes the configuration has infinite extent and no neutrons are lost from the configuration; and the effective multiplication factor, k-effective, which accounts for neutron leakage from the configuration. For a reactor to sustain a constant power its k-effective value must be unity. Reactivity is a parameter that expresses its deviation from criticality and is given by (k-1)/k.

The infinite multiplication factor of a fuel assembly is a strong indicator of the potential it has to produce power in a reactor. Grouping high reactivity bundles together in a reactor is expected to produce a higher power compared to a similar grouping of low

reactivity bundles. The bundle reactivity is therefore an important parameter for distributing fuel in a reactor.

As the fissile fuel mass depletes or "burns up" over use, fuel reactivity decreases. Used fuel is characterized by the number of cycles of core residence--fresh fuel has not been exposed, once-burned fuel has been exposed one cycle, and twice burned fuel has two cycles exposure. Bundles are chosen for a particular fuel enrichment, which specifies the concentration of U-235 loaded in the fuel. Both enrichment and exposure are factors that determine fuel mass any time during cycle life.

Neutron absorbers or "poisons" are placed in the core to limit the chain reaction and thereby control reactor power. They may take the form of control rods or burnable absorbers that introduce a localized reduction in power. Boric acid dissolution in the reactor coolant water introduces an equally distributed power reduction. The absorbing material of burnable absorbers diminishes with exposure and is incorporated with some fuel assemblies. Like dissolved boron, burnable poisons are introduced to offset long term reactivity changes due to fuel burnup.

Fueling strategies distribute fuel to satisfy one or more objectives such as maximum fuel utilization,

maximum cycle length, or minimum core leakage. How the fuel is distributed in a loading by any one strategy strongly depends on the available fuel reactivity.

II.B Conventional fueling strategies

The Out-In-In or simply Out-In fueling strategy requires all high reactivity fuel assemblies to be placed in the core periphery, the peripheral fuel to be moved to the interior of the core, the interior fuel to be moved to the center, and the lowest reactivity fuel near the core center discharged. In this manner, the reactor is fueled for regions that increase in reactivity radially toward the core center.

The second conventional fueling pattern, scatter or checkerboard loading, attempts to surround the faces of high reactivity fuel by fuel of low reactivity such that equal reactivity locations are at their closest diagonally, like the equal color squares of a checkerboard. A uniform distribution of fresh and old fuel throughout the core brings the local value of reactivity into balance.

Finally, a modified Out-In scheme exists whereby all fresh fuel assemblies are loaded in peripheral positions and once and twice burned assemblies are scatter loaded in the interior. These conventional loading schemes do not require burnable poisons to

maintain acceptable power distributions. Since no burnable poisons are used, the total reactivity of the core decreases monotonically with burnup. Therefore, the core reload pattern is normally designed to satisfy power peaking constraints near the beginning of cycle (BOC) since core reactivity is greatest then.

II.C Low leakage patterns

Loading patterns that place the high reactivity fuel closer to the core interior are known as low leakage loading patterns. Low leakage patterns, as opposed to conventional loading patterns, increase cycle length without increasing the number of fresh assemblies required. Consequently, low leakage patterns extend vessel lifetime by reducing fluence caused embrittlement. Fuel enrichment requirements are also reduced.

Two common low leakage schemes are the In-Out-In and In-In-Out patterns. The In-Out-In pattern places fresh fuel in the core central region at the outset, moves it to the peripheral region for the second cycle, then finally places the fuel back to the center for the third cycle. The second scheme places most fresh fuel in the central region of the core, moves the fuel from the central region to the interior for the second cycle, and the periphery is filled with fuel from the interior

for the third cycle. Since the peripheral assemblies in the In-In-Out scheme are more highly depleted than in the In-Out-In scheme, radial leakage is further reduced and less fuel enrichment is needed for the same energy requirement. The In-In-Out scheme also is characterized by a lower peak pin burnup and lower radial power peaking.

When low leakage patterns contain a high percentage of burnable poisons shuffling the fuel is no longer governed by power peaking constraints (Colman,1979). Peaking is controlled by burnable poison and shuffling of the burned fuel is utilized to obtain uniform burnup. Cores with increased use of fresh fuel containing burnable poisons are likely to exhibit a relative power increase near the fresh fuel as the poison depletes. Designing for BOC peaking limit, then, does not guarantee that peaking design constraints are satisfied throughout the cycle. The conventional BOC design practice becomes awkward when a low leakage strategy is required. Innovative strategies are needed when the local power peak to be minimized occurs not only somewhere over the position of the loading but also occurs somewhere over the cycle.

II.D Solution methods

In-core fuel management optimization methods may be

divided into two areas of problem solving--classical and direct search.

II.D.1 Classical optimization methods

A large portion of research on optimization of in-core refueling has used the classical problem solving techniques of operations research, namely linear programming. The classical methods of optimization generally cannot be used for in-core fuel management since the system equations are normally too complicated for manipulation. In addition, the component equations representing the objective function and constraints must normally be expressed linearly and to do so necessitates approximation and neglect of nonlinear interdependent parameters. Due to the time consuming nature of the linear programming optimization and computer memory space limitations additional simplifications must be applied. Nevertheless, many recent optimization methods have used various techniques including linear programming alone, linear programming and direct search, the gradient projection method, and Monte Carlo integer programming (Terney,1977; Mingle,1978; Chen,1977; Hobson,1986).

II.D.2 Direct search

Direct search in the core reload problem is an iterative procedure of loading pattern generation, evaluation, and selection that guides the solution by a set of rules to find the loading pattern which best minimizes (or maximizes) an objective function. In most applications, the objective is only to minimize the local power peak.

Since Naft and Sesonske, a number of researchers have used the direct search method (Naft,1970). Direct search applications later included rules to account for rotation of assemblies (Rothleder,1985;Robinson,1987). Direct search applications for low leakage fuel management remains a current research topic (Rothleder,1988). Hoshino was the first to equate the direct search method as the state space search of artificial intelligence (Hoshino,1972). Evaluating the solution performance by a chart, he recognized behavior of some rules always led to a reduction in the power peaking factor. The search could later be improved by enhancing the behavior. Research did not resume in this heuristic learning technique until recently by Galperin et. al. (Galperin,1989).

Aside from PWRs, the direct search method of core reload determination has been applied to BWRs and Liquid Metal Fast Breeder Reactors (LMFBRs) (Sekimizu,1978;

Lin,1979;Kobayahsi,1976). The primary difference in applications for these reactor types are neutronic so the same positioning guidelines are still used. However some additional restraints should be applied. Unlike a PWR, the large number of control rods in a BWR can easily adjust an adverse power peaking factor at BOC. At end of cycle (EOC), when the rods are almost fully withdrawn, this is not possible. Therefore limiting radial power distributions are normally encountered at EOC. Also the BWR has over twice as many fuel assemblies as a PWR so the combinations of trial moves become unmanageable. Since in a BWR the four bundles surrounding a control rod are deliberately matched for reactivity distribution, these bundles are treated as a block and used as a unit for shuffling. Once placement of the pattern is refined individual bundles are shuffled for further optimization.

II.E Other methods

Early algorithms in loading optimization required representation of problem components and constraints in a highly mathematical sense for numerical solution by linear programming methods. Presently, the symbolic representation features of artificial intelligence languages allow a more direct representation for components. Most recent and continuing research in in-

core fuel management attempts to dispense with mathematical representation altogether and to eliminate rule coding. This research uses techniques still under investigation in the domain of artificial intelligence, such as constraint satisfaction and neural networks (Dauboin,1989;Uhrig,1989).

II.F Typical placement rules

Many of the shuffling guidelines used in the past twenty years of research for in-core optimization are common to all procedures and were first reported by Kawai, Naft and Sesonke, and Stout and Robinson. They are, however, simple restraints familiar to the refueling expert (Kawai,1971;Naft,1970;Stout,1973). The guidelines normally are applied in phases that progressively improve the fuel distribution so that each move toward an optimal loading introduces a smaller overall change in reactivity. Many of the guidelines are only constraints that prohibit positioning. These may be summarized in four groups and are:

1. Central -- the center fuel assembly is not moved since it is unique and is treated separately.
2. Zonal -- regions are defined that must meet a specific bundle exposure or reactivity.
3. Symmetrical -- a power balance in the loading is maintained such that exchanges about the lines of symmetry are balanced.

4. Reactivity -- bundles are not exchanged if a bundle exhibits less power but higher reactivity to a second bundle.

A reasonable use of constraints can cause considerable reduction in search. By specifying one eighth core symmetry, a loading of four zones has 100 unique exchanges (Ho,1982). All the possible loading configurations could be tried within a reasonable amount of time and search would not be required. Application of too many constraints, however, could lead to a short sighted solution. The loading the search begins with dramatically affects solution performance as well. Details of the problem solving guidelines will be presented later and appear elsewhere (Stout,1973; Rothleder,1985).

A core power calculation is required to evaluate precisely the effect an exchange of two bundles has upon a loading. But often changing one bundle position by either exchange or rotation affects only the power distribution nearby. If the change in local power peak expected for a change in bundle position could be determined approximately, the worst moves that could potentially increase the local power peak could be quickly eliminated. To this end some researchers have used semi-empirical predictions in their calculations. For each potential move, Huang and Levine used the

following relation to estimate the power at a position where a move occurs from its four adjacent positions

$$P_{ij} = \frac{\rho_{ij}(W_{ij+1}P_{ij+1}+W_{i+1j}P_{i+1j}+W_{ij-1}P_{ij-1}+W_{i-1j}P_{i-1j})}{1 - \rho_{ij}(1 - 4W_{ij})}$$

(refer to Figure A.8) where

P_{ij}= power of position i,j.
W_{ij}= diffusion kernel of position i,j.
ρ_{ij}= reactivity (k_∞) of moved bundle in position i,j (Huang,1978).

Sekimizu used a similar equation and demonstrated this prediction is in remarkable agreement with a comprehensive two-dimensional power calculation (Sekimizu,1977).

Similarly, two-group perturbation theory may predict the power change following a move if the corresponding reactivity change is sufficiently small (Mingle,1975). Ho and Rohach assumed that if perturbation theory could be used and the reactivity change according to each trial move could be computed then the overall power peaking factor could be reduced by systematically choosing all exchanges that decrease the core k-effective (Ho,1982). Their research proposed such a solution which first ranked all assemblies in the core by decreasing reactivity then eliminated pairs of bundles from the list only keeping exchanges that decreased the overall reactivity.

Finally their new configuration was evaluated with a comprehensive power calculation. In actuality, perturbation theory works well only if minute changes of reactivity are expected but this is seldom true for any bundle exchange in a loading. Overall, it is best if simple power prediction methods are applied judiciously to eliminate the worst of moves and then followed by a comprehensive power evaluation.

III. MOTIVATION

III.A Justification for expert system

The nuclear power industry currently is in various stages of artificial intelligence integration for design, construction, operation, and maintenance. Expert system implementation for any given application has seldom been routine, however, since the number of diverse applications considered of AI outweigh our experience and computer tools. Often a problem cannot be formulated well enough for the solution desired to be possible, or it requires physical instead of cognitive skills, or the problem complexity is too unbalanced by implementation cost to be justified for an expert system. Therefore, we will first review whether an expert system solution to the core reload problem is possible or justified.

III.A.1 The problem is time intensive

A shortage of time often limits the effort that can be devoted to the optimization. Although the reload procedure is planned months in advance, the core reload engineer must respond quickly to unanticipated changes in the design. For example, fuel bundles intended to operate through the following cycle may, upon inspection, be damaged or become damaged during the

shuffling process. Preceding maintenance, normally scheduled on the day the reactor is shutdown for refueling, may run behind schedule and force refueling operations to respond quickly to make up time.

III.A.2 The task has high payoff

With shutdown expenses about $500,000 per day, the time required for reactor maintenance and refueling is costly. The computer codes used to evaluate each proposed loading are themselves time consuming and expensive to operate, and must be used extensively. An optimal reload solution would minimize labor and computer run time but maximize fuel use and power generation for the cycle. Considering that all nuclear utilities must once a year refuel their reactors subject to their own constraints and preferences, a common refueling tool would benefit all of them.

III.A.3 Human expertise is in short supply

Human expertise in the highly evolving and complex nuclear discipline has always been scarce, consequently there is a strong dependence on overworked experts. As fuel management experts reach retirement age, their substantial expertise will be unavailable tomorrow. Also, as advanced fuel management technologies or

methods evolve, greater dependence is placed on those few experts familiar with the applications. An AI application to the core reload problem, therefore, is highly desired to retain and distribute their knowledge.

III.A.4 The core reload problem requires automation

Designing loading patterns by manual optimization is an iterative process prone to human error. The engineer performing the design work must repeat many of his decisions and interactions several times over. Even an experienced engineer may make mistakes and arrive at a reload pattern far less than optimal. The elimination of manual steps leads to fewer potential sources of human error, a reduction in human expertise required to direct an iterative problem solving process, and allows a rapid response in the design when unforseen situations occur.

III.A.5 The task requires extensive knowledge

The number of distinct loading patterns a given reactor may assume is given by

$$t_n = n!\ p^n$$

where

n = number of assemblies

p = number of rotation positions

t_n = number of unique loading patterns.

When only considering the position and rotation of bundles filling a quarter core (n=56) the total number of possible patterns is on the order of 10^{118} !. The optimal solution could never be found from these many combinations. It is possible, however, to formulate a problem solving methodology to direct the search and constrain the number of alternative patterns considered so the method, although no longer guaranteed to find the best answer, will always find a very good answer.

Integer programming and similar weak search methods have been feasible in the past for the core reload problem (Haq,1985;Comes,1986). Such schemes normally systematically explore the loading resulting from each trial move. The process proceeds without the physical intuition the expert possesses to lead the search to an optimal loading in the least amount of time. Whereas any modern means of programming can solve the requirements listed above it is the ease of introducing knowledge to direct search that the tools of AI are most specialized.

III.B Advantages of AI solution methods

In the early seventies, AI researchers in their quest for incorporating intelligence to computer

programs focused research on developing techniques in representation and search. It was not until the late seventies that the AI community realized that the problem solving power of a program originates from the knowledge it possesses, not just from the inference schemes and formalisms it employs. The use of extensive, quality, specific knowledge about a narrow problem area made the problem solving program intelligent. These programs were called knowledge-based systems. Those knowledge-based systems that were given the capability to explain their own reasoning were known as expert systems.

An expert system solves a problem in the same manner as an expert since it embodies his expertise. The expert system user, on the other hand, may not be an expert but only needs to be familiar with the problem domain. The system makes the decisions or suggests the decisions and explains its conclusions. Although many expert systems are written in traditional languages, the goal of expert systems development attempts to represent the expertise explicitly and provide a basis for explanation, as opposed to "compiled" expertise into FORTRAN variables and statements.

Many core reload expert systems that exist use heuristic search methods that were in use during the representation and search generation of AI research.

Since then a new generation of computers and software has become widespread. Problem-oriented languages intended to solve calculations, such as FORTRAN, adapt poorly for work in artificial intelligence since few complex concepts may be realized mathematically or programmed as simply as with symbolic-manipulation languages (e.g., LISP). A substantial amount of time is required to formulate carefully knowledge into programs written in traditional languages and the coding becomes progressively inscrutable and difficult to maintain as knowledge is added.

Recently, commercially developed AI software development environments have become available that provide a kit of software tools designed to assist building knowledge-based systems. The tools attempt to remove the requirement of specialized programmers to translate expert knowledge to code and interpret output. These building tools can provide explicit representation of symbolic structures, behavior, and reasoning so that the structures can be examined and reasoned with (as opposed to buried in code). They may also provide specific problem solving paradigms for implementation. Instead of restrictions to computer printouts for survey of results or program listings the new environments may provide graphic aids to present to developers and users the representation and reasoning of the expert system.

Useful graphic presentations include schematic diagrams of models, trees of rule connections, analysis displays of graphs and charts, and images that the user can directly interact with to modify parametrically the model. These graphic direct interactions and presentations allow the developer to interact with the system and experiment with the problem.

III.C Shortcomings of AI solution methods

The lack of comparable speed and ability to perform numerical calculations may be the only disadvantages of implementing knowledge-based systems with the current AI tools as opposed to traditional compiled languages. AI languages have traditionally emphasized capability with symbolic structures and began as research tools outside engineering applications. They overlook calculational capabilities required for engineering applications or a simple interface with conventional languages that have them. Development of both improved software and specialized computer architectures to manage AI symbolic languages with improved speed are now independently being pursued. Commercial AI software development is generally moving toward applications which run and may be imbedded in standard computing environments. The current emphasis upon implementing most applications in C improves efficiency in execution speed and memory

utilization as well as interface since most conventional software is written in C (Stone,1987). Alternatively, instead of waiting, once the reasoning and representation of an expert system is finalized using environment tools a version compiled in a traditional language could be implemented for production purposes.

III.D Tool selection issues

Choosing an appropriate tool for building an expert system is one of the most difficult decisions to make in expert system development since most tools were developed to handle a particular class of problems. Many tools were adapted from earlier research systems after stripping the systems of their knowledge (e.g., EMYCIN) and other more recently developed tools incorporate what their developers hope will enable the tools to be appropriate for more types of problems. Ease of understandability for the user is often sacrificed when the tool's creator adds to its capabilities; a loss of function occurs when the tool is simplified. Most existing expert system tools commercially available are unsuitable in at least one respect to any given task and it is important to understand initially the nature of the problem to be solved before selecting a tool and implementing it.

IV. PROBLEM FORMULATION

The difficulty of starting on a knowledge based system appears to be not with understanding search and inference strategies but selecting tractable representations for the task from a universe of real world knowledge. A knowledge based system may contain a large amount of domain dependent knowledge to cope with special problem solving situations yet is a weak problem solver because of an unmanageable assembly of conceptual exceptions and redefinitions built on a problem formulation which is incorrect or incomplete (Ernst,1983). The most effective problem solving methods are developed given a concise problem formulation. Solving a problem therefore begins with the problem formulation itself. Real world problem solving, of course, involves a lot of problem reformulation but stating the problem in systematic form leads to an understanding which reduces this task.

Possessing an explicit model of a difficult problem solving process is itself a benefit. Coding expert knowledge, normally elusive and implicit, is a revealing process which lends to many improved insights within a particular domain. Experts may gain a significant amount of experience in their field, but they are often unable to document this. Therefore, in some cases, the knowledge gained in order to build the expert system is

even more valuable than the actual finished product. Not only does the expert system make the expertise available to non-experts, but the expert's heuristic knowledge and problem solving strategies are explicitly documented and made available to others for study and examination.

A structured analysis aids in selecting an initial problem formulation by separately defining the objects, operators, and scheme of the solution. Independent investigation of each component of the problem fosters an efficient representation and reduces chances of introducing inherent conflicts. As a gradual approach to hard problems, the structured analysis builds representations from the bottom up following the natural structure of the problem domain. This section presents the formulation for the core reload problem stated in three parts: the problem domain components, operators, and cognitive aspects.

IV.A General problem solving knowledge

Before defining the objects in the formulation, we will first review the information the fuel management expert finds important to solve the core reload problem, the representation to record solution history, and the responsibilities the search strategy has for control of the solution.

IV.A.1 Required information

When the fuel management expert approaches the core reload problem, he recognizes the core should be divided into several zones and in each zone should be loaded a fuel bundle with a reactivity of a desired range. Matching bundle reactivity to zones attempts to obtain a power balance in the loading from center to periphery. Each loading investigated in the search must also have the same exposure and boron concentration if power peaking is to be compared. This requirement ensures the reactivity of each bundle is constant at any time.

The expert sets out upon the core reload search intending to satisfy a search objective. This objective usually is to minimize the loading local power peak but may include other objectives. At any point during the search, the expert requires information regarding the loading under investigation in order to evaluate it, apply heuristics to decide which move to implement next, or decide which direction the search is to proceed. This information includes

1. The reactivity for each bundle positioned in the loading.
2. Regions of the loading which require special attention as to bundles placed in them.
3. The loading power profile, contents, soluble boron concentration, and period in the cycle.

4. A history of all loading patterns attempted in the search.

5. The best loading obtained so far in the search.

6. Moves already considered for a loading, or, moves which may be considered.

7. The strategies already applied in the search and the strategies still available.

As discussed above, items one and two of this list normally remain consistent throughout the solution. Each unique loading is characterized by a different power profile and therefore item three changes with each new point of the search. Finally, items four through seven are determined by all loading patterns so far generated. Therefore, the information the expert needs throughout the solution may either not change during the search, originate from a single loading, or depend on all loadings generated by the search. It is the information that changes as the solution proceeds that must be recorded to be able to analyze if the solution is converging. If the search must backtrack, information must also be recorded to be able to restore and continue the search from a loading pattern generated previously.

IV.A.2 Solution state space and traversal

There is no definite solution to the core reload

problem; an acceptable loading is determined by applying heuristics and investigating many trial loadings. The history of the problem solving search as it proceeds from an initial loading to each succeeding loading configuration may be imagined as a graph containing nodes corresponding to states (Figure IV.1). The nodes of this state space graph are linked together by arcs that represent the moves that transform one loading to another (Figure IV.2).

The expert applies moves to a particular loading and successor loadings are generated until the search shifts to a successor which satisfies the search objective best. Moves may be classified as either partially commutative or non-commutative:

1. Partially commutative -- moves which will improve the final loading no matter what sequence they are applied.

2. Non-commutative -- moves which when performed out of sequence will certainly lead to a poor final loading.

Moves of class one may be generated for a particular loading but applied to any succeeding loadings. Such moves introduce a small change in reactivity to the loading and normally do not change the position of the local power peak but tend to reduce it. Moves which rotate bundles are the most common form of partially commutative moves. Non-commutative moves are specific

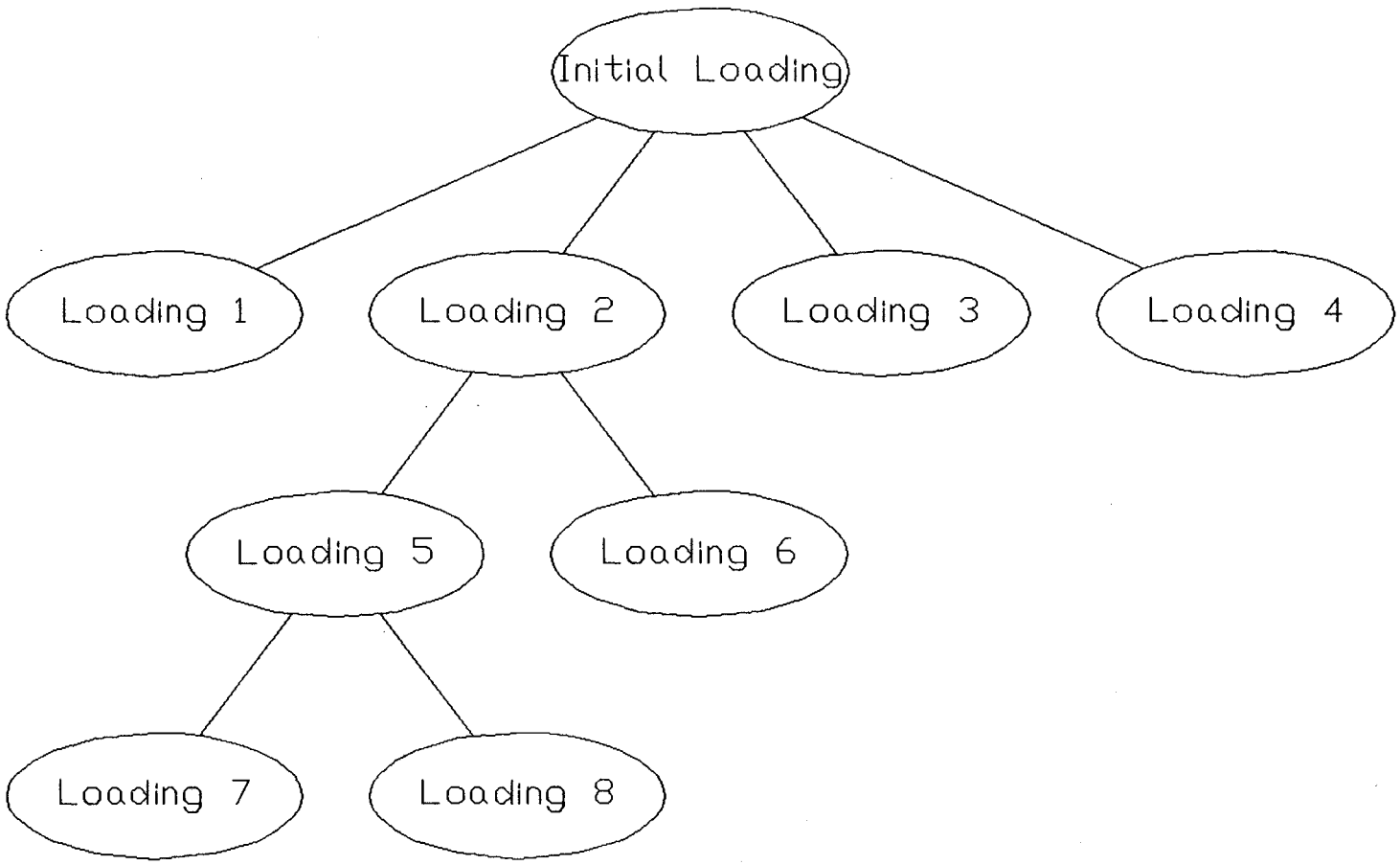

Figure IV.1. Sample state space of core reload problem.

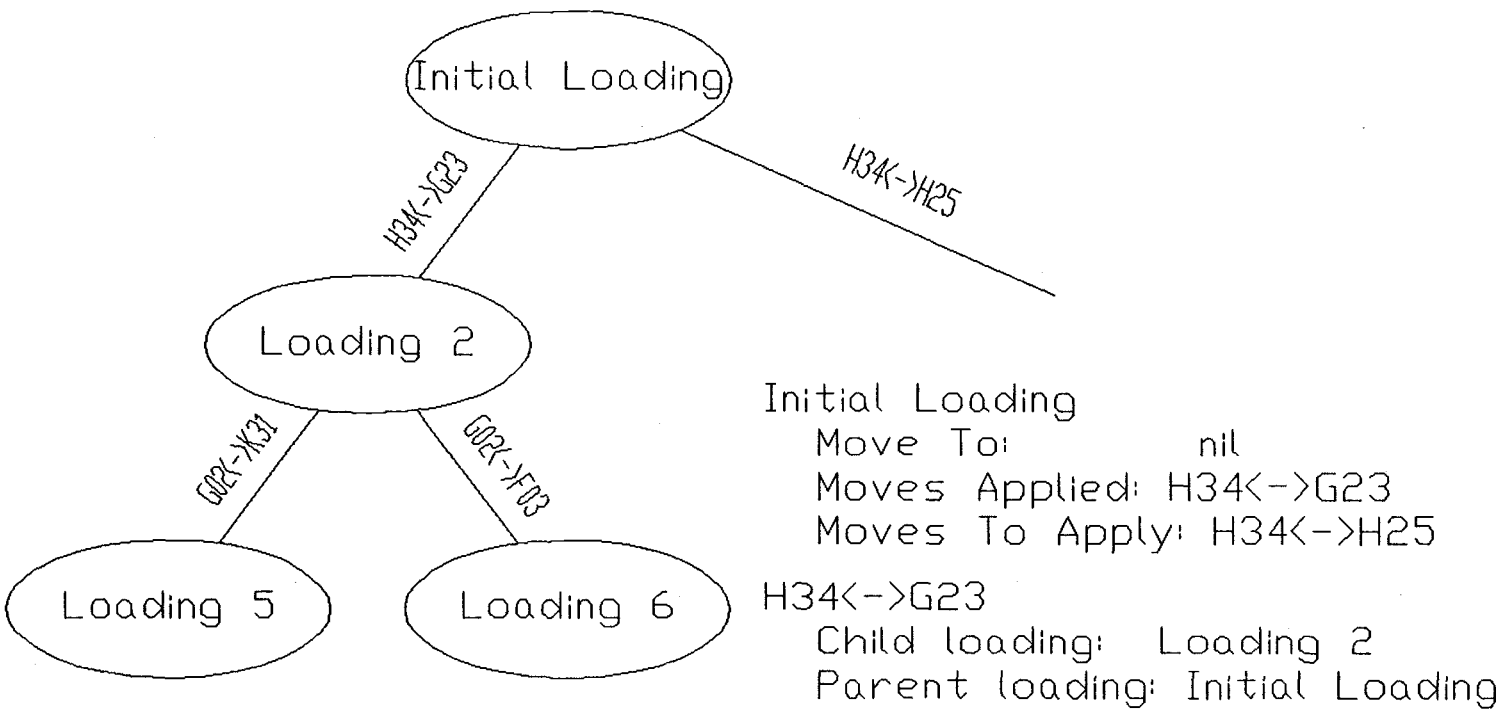

Figure IV.2. Relationship of loadings and moves in state space.

to the loading they are generated for and, with few exceptions, cannot be applied to succeeding loadings because they increase the local power peak.

For each loading, either moves may be generated one at a time as each new successor is required or many moves may be generated in advance then one move is selected from a list as needed. There are a few incentives to generate as many moves at once for a particular loading as can be obtained as opposed to generating each move before use. First, when heuristics are applied to generate moves some calculations are performed which may be unnecessarily repeated in subsequent move generations. Second, the requirement to verify that unique moves are generated for each loading is eliminated. If it is decided to generate a list of moves, the strategy must choose from the list the best move for the time applied. Alternatively, the list may be ordered at the time moves are added so that the best moves will be withdrawn first.

As the solution continues and the state space increases it becomes important for the problem solver to avoid a move which could recreate a prior loading. This could require the time consuming process of checking the proposed loading configuration against all loading states previously generated by cycle exposure, boron concentration, and bundle position and reactivity.

Instead, the loading power peak value or k-effective could be used to distinguish each state since it is a strong function of the loading configuration itself when taken to an adequate number of significant digits. Since an objective is to obtain progressively lower power peaks as the solution proceeds and an extremely large number of orientations is probable it is very likely only unique loadings will be accepted. Therefore we are assured new loadings are used for the solution without verifying uniqueness of each loading configuration. Care must be taken for a particular loading to ensure each move tried is distinct. Recording each move applied in a loading is a simple process, however.

IV.A.3 Responsibilities of strategy

The search strategy directs the solution so progress is made toward finding a loading pattern that best satisfies the objective function. Based on the above discussion, it has the following requirements:

1. Generate unique moves either one or several at a time.
2. Add and order the moves to apply on a list.
3. Choose the best move from the list for implementation.
4. Evaluate the new loading generated and decide which loading to continue the search.

In addition to traversal, the strategy includes the expert's heuristics to generate moves for a loading. These heuristics will be covered in a later section.

IV.B Problem domain components

This first section defines the objects, their attributes, and their important relationships in the problem domain. For a concise formulation, each object within the domain is unique but the number of object instances which appear in a solution, either at once or throughout, is defined during the problem solving process. The restrictions on this number is given for each object below. Also, a means of grouping instances within a list is necessary to express some association between them and with what they may belong.

A hierarchy of objects is implied such that parent objects are composed of subobjects. Each object has distinct properties which may be inherited by its subobjects. For example, a configuration of a core is a loading and a loading is composed of bundles (Figure IV.3). A bundle has reactivity, exposure, power and location when placed in a loading. A bundle inherits the boron concentration of its loading. Objects for the fuel pools are included for completeness since these are the locations fuel bundles may be discharged from or

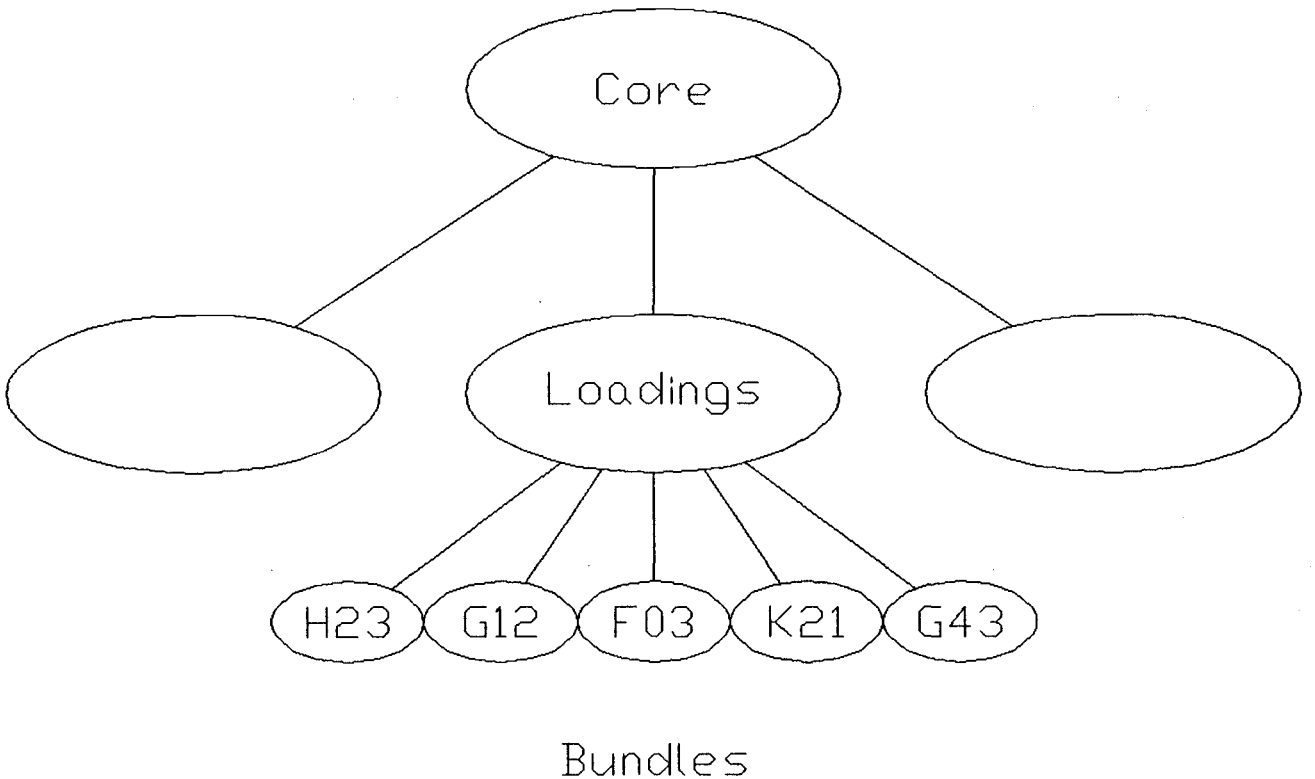

Figure IV.3. Hierarchy of objects.

obtained for a loading.

Core

Knowledge regarding the reactor design and operation history. This information remains consistent between cycles and is normally implicit to the expert's strategy. There must be one instance of core per solution.

Cycle	The cycle name (e.g., cycle 6)
Exposure	Cycle exposure at time of investigation.
Type	The class of reactor (e.g., PWR or BWR).
Capacity	The number of bundles the core holds.
Boundaries	The location of characteristic regions in the core, such as the core centerlines, periphery, quadrants, or lines of symmetry.
Safety Constraints	The limiting key safety parameters which define the operational range of the core and determine an acceptable loading.
Economic Constraints	Specifications which when met determine an optimal cost loading.

Loading

The proposed core configuration for a given cycle is characterized by choice of bundle type and position as well as poison concentration. The representation is

simplified if the loading is conveniently defined as a quarter core. Note depletion and life attributes are unnecessary for loading patterns accepted by BOC calculations. There may be any number of trial loadings in a solution but there must be one initial loading.

Three attributes given to a loading determine the moves to other loadings in the state space: a move to the parent loading, a list of all moves to successor loadings, and a list of moves to produce new loadings.

Move To	The move performed to obtain this loading.
Moves Applied	A list of moves applied to obtain successor loadings.
Moves To Apply	A list of moves to apply to generate successor loadings.
Contents	A dictionary of bundles which fill the quarter core loading indexed by map position.
Keff	The effective multiplication factor of the loading.
Shim	The boron shim concentration in ppm.
Exposure	The loading exposure (GWD/MT).
Maximum Power Bundle	The bundle of highest power in the loading.

Move

A move represents an arc between trial loading nodes in the state space search and identifies those bundles that change position from a loading to a

successor loading. Moves exchange, rotate, or interchange from a pool the bundles in a loading and may also specify the loading exposure or boron concentration.

As in the STRIPS methodology, two lists represent the change in state by a move: a delete list containing those bundles of the loading prior the move whose characteristic properties are "deleted" and an add list of bundles with properties distinct to the resultant loading (Fikes,1971). Any bundle not in the add or delete list does not change between loadings. To generate a new loading, the bundles in the delete list are removed from the current loading and the bundles in the add list update the loading. Alternately, to revoke a move the bundles in the add list are removed from the resultant loading and the contents of the delete list replace them. This operation conserves memory since only those bundles which change between loadings need be represented at each state.

Name	The name of the move identified by names of the bundles to swap (e.g. H34<->C23).
Add Bundles	A list of bundles to be added to the current loading to generate the successor loading when the move is applied.
Delete Bundles	A list of bundles to be deleted from the current loading to generate the successor loading when the move is applied.

Child Loading	The loading generated when the move is applied.
Parent Loading	The loading to which the move was applied.

Bundle

Knowledge representing a single fuel assembly. Bundles, particularly those located near the periphery, tend to burn up unevenly if a flux gradient exists across it. To model bundles which may be exposed to steep burn-up gradients, the power, reactivity, and exposure of each bundle is identified by corner. The bundle representative value is averaged from all corners. Note position, power, and rotation only have meaning when a bundle is placed; all remaining parameters are bundle physical characteristics.

Name	Bundle identification.
Exposure	The current burn-up by corner.
Reactivity	K-infinity value by corner.
Position	The grid cell position in the loading where the bundle is placed.
Rotation	Bundle rotation as an integer from 1 to 4. A value of 1 corresponds to the position of the initial loading and 2, 3, and 4 each represent progressive 90 degree clockwise rotations.
Power	The relative power by corner.
Cross Section Library	An index unique by bundle type to

reference nuclear cross section correlations as a function of exposure.

Pool

A pool is a holding place for fuel bundles outside the core where bundles may be discharged from the core or selected for loading in the core.

Holdings List of bundles in pool.

IV.C Operators

Whenever an object instance is created, deleted, moved, or its attributes change an operator has acted on it. There may be any required number of operators but the fundamental operators which appear here seem to be the most complicated to implement. In addition to the operators mentioned here some common set theory operators must be available to perform intersections and unions of lists. An ability to collect the operators in a procedure is required.

Exchange(bundle1, bundle2)

Exchange the positions of bundle1 and bundle2 within the core. Different actions are taken based on the power symmetry that should be maintained about the loading. The rules are the same if one quarter core symmetry or one eighth core symmetry is imposed. If any

bundle is on the line of symmetry then choose one of the following responses to maintain a symmetrical power distribution:

1. When one bundle is not on the line of symmetry and the other bundle is then place in its transpose position a replica of the first bundle when the exchange is performed

2. When both bundles appear on or between lines of symmetry then exchange the bundles which appear in their transpose positions as well.

Neighborhood(bundle, domain, extent)

Return a list of all bundles from domain which surround the named bundle in the loading by distance extent. This operator is applied to determine the set of bundles which the bundle locally influences.

Power(loading)

Compute the power statistics for the loading.

Rotate(bundle, degree)

Rotate the named bundle in its current position in 90^{o} increments by degree.

Insert(bundle, origin, destination)

Remove the bundle from the origin list and place in the destination list.

Remove(bundle, origin, destination)

Performs opposite task of Insert operator.

Select(domain, conditions)

Return a list of all bundles from domain which satisfy conditions (e.g. select all bundles of exposure greater than 20GWD/Mt).

Reject(domain, conditions)

Return a list of all bundles from domain which do not satisfy conditions (e.g. reject all bundles on the periphery of the core).

Sort(domain, conditions)

Return a list of elements from domain sorted according to the declared conditions (e.g. list all bundles in order of increasing reactivity). Sort could also be used to determine the bundle with some minimum or maximum value if the head or tail of the list is returned.

IV.D Cognitive aspects

The strategy to choose and apply operators embodies the cognitive aspects of the domain. At least an algorithm or search scheme is required but heuristics and rules must be included to provide knowledge. When

probabilities are understood between alternate paths the cognitive formalism may include levels of confidence in its reasoning. An ability to iterate a procedure until some condition is met must also be provided.

IV.D.1 Initial placement rules

Before the solution search begins, heuristics are applied to generate an initial loading configured to balance the reactivity throughout the core with the new fuel. Since the solution is strongly dependent on the initial loading configuration, this is the important first step.

Rank bundles according to reactivity and place:

1. Highest reactivity fuel in periphery region.
2. Next highest reactivity fuel in the interior even (odd) region. This will include some new fuel assemblies.
3. Lowest reactivity fuel in interior odd (even) region.
4. Remaining fuel in the intermediate region.

IV.D.2 Solution method

The problem solving method for the core reload problem employs heuristics to direct the search and constraints to limit the number of configurations considered. These heuristics and constraints applied on

top of a general-purpose search method provide the intelligence of the search. Although the hill climbing method is used here, the search could use the more complicated best first or branch and bound search methods (Pearl,1984).

Search method

This search method is known as the hill climbing technique and is classified as a weak search method (Pearl,1984). The initial loading is named the best loading when the search begins.

1. Generate a move. If no new moves can be obtained begin the next subgoal or report failure.
2. Implement the move and create a new loading.
3. Calculate the power distribution of the loading, and
   a. If the objective function value for the loading is less than the best loading value then record the current loading as the best loading and continue.
   b. If the objective function value for the loading is greater than the best loading value then revert to the best loading.
4. Continue with step 1.

Position constraints

Investigation of the problem solving method revealed that most bundle placement constraints, like the search method, remain common throughout the problem solution and may be classified as either single bundle or exchange constraints.

Single bundle constraints

1. Bundles located on the periphery are never moved.

   Reasoning: Bundles of the highest reactivity are placed in the periphery, the region of highest leakage. Moving a peripheral bundle into the interior will cause a power peak about that bundle.

2. Fixed bundles are never moved.

   Reasoning: Bundles may be declared fixed in the core by the user.

3. Bundles with an insignificant reactivity gradient are not rotated.

   Reasoning: A relevant change in power is not likely compared to rotation of bundles which possess significant burn-up gradients.

Exchange constraints

1. Bundles in the intermediate regions may be exchanged with fuel in any other position except the periphery.

2. Even parity bundles may not be exchanged with odd parity bundles.

   Reasoning: Scatter (checkerboard) loadings produce flatter power distributions than zonal loadings. Deviating from the pattern is expected to produce undesired power peaks. The peripheral, interior even (odd), intermediate, and interior odd (even) regions have fuels of decreasing reactivities initially assigned to them. Placing matching reactivity bundles next to each other, particularly within the core center where fuel placement is more critical due to low leakage, is likely to upset the power balance in the loading.

3. High reactivity bundles should never be moved an extent more than one position during one move in the reactor interior.

   Reasoning: High reactivity fuel assemblies placed in the reactor interior cause large perturbations on the local radial power peaking. Therefore, improved results are expected in the shuffling

iteration scheme if these elements are moved only one position at a time; then a new power calculation is made to determine the next move, rather than moving the bundles directly from a high power area to a low power area.

IV.D.3 Problem heuristics

The core reload problem has been divided into three phases, each phase positioning progressively less reactive fuel by unique move instructions. If moves referenced to a particular set of bundles fail to reduce the radial power peak an alternate heuristic is applied to find moves over a less specific region of the loading. When some serendipitous configuration is discovered, heuristics are also applied to direct the search away from its normal course to investigate new moves. The heuristic search strategy for each subgoal is composed of such rules.

IV.D.3.1 New fuel strategy

(Rules to redistribute most reactive fuel)

Move instructions

Given:

a. The maximum power bundle which satisfies single bundle position constraints.

b. A local minimum bundle which is defined as the minimum power bundle from the neighborhood of extent two about the maximum power bundle.

Consider for exchange all bundles which satisfy exchange constraints referenced to the maximum power bundle and are nearest to the local minimum bundle (one to at most two exist). Consider all moves.

Heuristic search strategy

Apply each rule until it fails (i.e. all moves are applied) before attempting the succeeding rules.

1. Generate moves for the maximum power bundle towards its (1st) local minimum bundle.

2. Generate moves for the second highest power bundle towards its (1st) local minimum.

3. Generate moves for the maximum power bundle towards its next (2nd) local minimum.

4. Generate moves for the maximum power bundle towards the third highest local minimum.

Apply this rule whenever the power distribution is calculated. This overrides the search strategy to allow investigation of moves located in the other half of the quarter core loading.

If the new maximum power is greater than the best loading maximum and the new maximum is in the transpose position of the maximum of the best loading, then continue for the next step with this loading.

Apply this rule whenever moves are considered.

If a bundle to be exchanged has a greater reactivity than the maximum power bundle then reject this move but consider instead the exchange bundle as the maximum power bundle and generate moves.

IV.D.3.2 Old fuel strategy

Rules to redistribute less reactive fuel

Position Constraints

For this strategy add this constraint to the general position constraints. In the last strategy this rule was used in the heuristic search strategy.

A bundle is never exchanged with another bundle possessing a lower reactivity and greater power.

Reasoning: This exchange would cause the maximum radial power peak to increase since it would bring more reactivity into an area which already has high power.

Move instructions

Given:

a. The maximum power bundle.

b. A local maximum power bundle which appears in the neighborhood of extent one from the maximum power bundle and satisfies single bundle position constraints.

Consider for exchange the minimum power bundle of all bundles which satisfy exchange constraints referenced to the local maximum power bundle. Consider all moves but select so the move with the bundle of least power is first.

Heuristic search strategy

Try generating moves by each succeeding method until one method fails or all eligible bundles in the neighborhood of extent two have been tried for exchange.

1. Generate moves for the (1st) local maximum power bundle and any eligible low power bundle.

2. Generate moves for the second highest local maximum bundle and any eligible low power bundle.

3. Generate moves for the local maximum power bundle in neighborhood of extent two about the maximum power bundle with any eligible low power bundle.

IV.D.3.3 Rotate fuel strategy

(Wang,1987)

Move instructions

1. Given:

   a. The maximum power bundle.

   b. A list of all bundles in a neighborhood about the maximum power bundle which satisfy single bundle position constraints.

   Consider for rotation each bundle of the list above. Use rotation heuristics to order moves which will most significantly decrease the power peak first. Consider all moves.

2. Rotate the bundle 180^{o}.

Heuristic search strategy

Apply each rule until it fails (i.e. all moves are applied) before attempting the succeeding rules.

1. Generate moves for rotation of bundles in the neighborhood of extent one from the maximum power bundle.

2. Generate moves for rotation of bundles in the neighborhood of extent two from the maximum power bundle.

As an alternate source for rotation heuristics, Rothleder has recommended the following rules (Rothleder,1986). These rules are more difficult to implement since they require reactivity comparisons to be made from each corner of a bundle to its neighbors. For each surrounding bundle this requires eight

reactivity comparisons before a judgement can be made.

Rotation heuristics

1. Arrange the highest burned corners of those bundles surrounding the bundle corner with the radial power peak towards that corner.

   Reasoning: Adjusting corners of bundles surrounding the bundle with the radial power peak such that corners with lower reactivity than that with the peak should reduce the power peak.

2. Arrange the lowest burned corners of the bundles far from the bundle corner with the radial power peak towards the center of the core.

   Reasoning: This avoids directing gradients toward each other and adjacent placing of high reactivity corners.

IV.E Summary

The objects, operators, and heuristics above comprise an example formulation of the core reload problem for easy solution. The next section discusses the desired techniques of knowledge representation and appropriate support facilities to implement the core reload problem.

V. REPRESENTATION

The previous section stated the real world objects, the operations, and the problem solving knowledge necessary to solve the core reload problem. What is needed now is a concise expression of the problem components, relations, and operations in a symbolic form for easy solution by the problem solving tools we choose. Due to the symbolic representation common of AI tools, the representation required has already been simplified by choosing an AI solution instead of a strictly mathematical solution such as linear programming. Two representation schemes therefore require definition--the knowledge representation for concepts and the spacial representation for bundles in the map.

V.A Knowledge representation

More than one means of representing the knowledge of the problem exists. The representation methods include rules, object-oriented, frame based, procedural or other similar representations that combine these methods.

V.A.1 Rules

Rule based knowledge representation employs

statements called rules that typically have the form of If condition Then action. In this form rules specify the heuristic recommendations, strategies, or directions of the problem solving process. Rule based systems use an inference engine that interprets from which rules to infer new knowledge and decides the order in which they are applied. When a rule is interpreted and all conditions of the If part of a rule are satisfied by the state of the problem, the action specified by the Then part is performed. This action may cause an external interaction with the system (e.g., input/output), instruct the system to arrive at a conclusion, or change the problem state and thereby trigger other rules to be applied.

There are two approaches inference engines may apply to select and order applications of rules--forward chaining and backward chaining. The difference in the two hinges on the method in which rules and data are searched. Backward chaining or goal directed reasoning begins with what it needs to prove and executes only rules relevant to establishing it. Forward chaining or data directed reasoning first matches each rule's conditions to the state of the knowledge base to establish if a rule applies then executes those most specific.

Using rule-based systems to encode problem solving

knowledge has these advantages:

1. The If-Then form of rules are best capable for coding the situation-action reasoning experts tend to express themselves in.
2. When rules are at an appropriate level of detail they may be used to explain conclusions of a solution by retracing their actual lines of reasoning and translating the logic of each rule employed into natural language.

However, many rule-based system implementations have limitations that make them difficult to use for applications requiring iteration or backtracking. The highly specific knowledge of the expert reduces a large search space to the small search space of a specialized knowledge-intensive program. Many knowledge based systems such as EMYCIN work for a progressively refined specification of the problem until all evidence ensures a given response. Search is more of a last resort for these problem solvers. Therefore, an iterative generalized problem search method has not been adopted in most rule based systems. Some rule based systems allow rules to fire repeatedly provided the problem state matches the rule conditions (e.g., OPS5, PROLOG), but, since iterative search was not anticipated by their developers, most process a single rule once.

Rule based systems fail to represent all forms of

problems well. Most commonly do not provide an adequate means for grouping classes of rules that pertain to specific problems so that rule sets are more manageable. The If-Then structure of rules cannot pose an exception to prohibit an action of a default procedure such as "if A occurs, then do not perform B" or choose one of a selection of actions as "if A occurs, then perform one of the following..." Finally, the expressive power of rules is inadequate for defining terms or describing objects and their static relationships within the problem domain.

For the core reload problem, the heuristics and constraints found in the cognitive formulation presented above are a natural target for a rule based system that allows iteration of rules. Each statement, after some corrections required to express rules in a form an expert system may interpret, could be represented, and the reasoning text provided with each rule could be accessed by an explanation facility.

V.A.2 Objects and Frames

The distinction between frame and object representations has become blurred by the implementation of current AI languages that provide them. In time they may become synonymous and perhaps only the term "object" will be used to refer to both just as the term

semantic network seems to have fallen from use and is now understood as a form of frame representation. Either provide a structure for a knowledge base that is hierarchically ordered. A concept in such a knowledge base is represented by an object or frame and all its relevant information is defined by the object's attributes or the slots of a frame. Procedures attached to an object, known as methods, define its behavior and are activated by a process called message passing. Likewise, frames have procedures associated with slots that are invoked when data in the slot changes. The hierarchical relations between objects allow one to pass attributes, their values, and methods from a parent object to a child object. Therefore, a considerable economy of representation is possible by using objects or frames.

Object and frame based methods are attractive because they provide a concise representation of useful relations in a structured knowledge representation. The problem domain components of the core reload problem presented earlier are such a structured representation and is best suited for object-oriented programming.

V.A.3 Procedural Methods

Subroutines written in traditional programming codes are appropriate if the task performed is routine

and defined well enough to be encoded in a standard language. Procedures may be incorporated to perform a specific task in a knowledge based system, or, by nesting procedures under a set of high level control procedures, comprise the knowledge based system itself. The use of compiled procedures in traditional hardware and software allows high speed features of knowledge based systems to be available on many machines and be directly implemented with common programs and databases. Procedures provide great flexibility to the expert system builder since he is not restricted to use the control scheme defined by a given inference engine or reside within the specific class of problems an expert system tool was built for. Interest in implementing knowledge based systems strictly in procedural languages continues because of these payoffs (Butler,1988).

Simply developing a knowledge based system in a procedural language avoids the ease of updating and maintaining a knowledge base and the ability to explain results that expert system tools were meant for. Programmers as a rule are also required for the coding. Full implementation in a procedural language is only practical once the knowledge based system is finally developed. If the problem to be solved requires symbol manipulation, automatic memory allocation, and a uniform treatment of program code and data then

traditional procedural languages such as FORTRAN are inappropriate and instead AI languages such as LISP are recommended. In turn, LISP is poor in performing the numerical calculations FORTRAN is best for.

Some means of reaching an appropriate procedural language from a knowledge based system is therefore understandable. Knowledge based systems which encourage users to access the underlying system language, provide facilities so that access is direct and relatively easy, and interface to other procedural languages specialized in numerical calculations or databases are most flexible. Such knowledge based systems are known as having an open architecture.

The operators of the core reload problem are best implemented as procedures. Although object-oriented representations must support methods or procedures to define actions between objects, the object-oriented languages do not usually possess the numerical capabilities sometimes required and found in traditional procedural languages. Procedures can seldom be implemented using a rule based language.

V.A.4 Hybrid systems

Many AI researchers claim no single currently existing representation can model the generality of real world knowledge and therefore support a hybrid reasoning

scheme (Takenouchi,1987;Kunz,1984;Fikes,1985). A hybrid reasoning scheme unifies all methods of representation, particularly the models of skills of a rule based system and the patterns for describing and recognizing recurring sets of an object-oriented environment. The synthesis eliminates the shortcomings of using just rules or objects in representation. All individual representations become objects or belong as attributes in the scheme. For example, rule based systems are created as objects in the hierarchy and known as knowledge sources.

A hierarchical structure of knowledge sources provides an efficient partition of rules by class--something a pure rule based system lacks. Knowledge sources may be organized to a class-subclass taxonomy whereby each source contains only those features that distinguish it of more general super classes. The hierarchy acts as a discrimination network for successively refining the classification of a given object to satisfy the preconditions of a rule. In effect, the problem structure defined by an object-oriented network serves as discrimination between rules rather than a long list of conditions in a rule's If statement or discriminated by a chain of rules (Fikes,1985;O'Hare,1985). It can be argued that a pure rule based expert system can require over 3/4 of its

total rules to express declarative knowledge (i.e., hierarchical relationships between objects and their attributes) (Kunz,1984). The knowledge base becomes too opaque to the users and a detriment to system performance if 400 or more rules must be used to define what 75 hierarchically ordered rules represent in an object-oriented environment.

Hierarchical structuring of knowledge sources as objects provides a convenient way of selecting from multiple rules. If the hierarchy is interpreted as a hierarchy of authority, then the rule of highest authority is selected. If the hierarchy is interpreted as one of specialization, then the more specialized source of knowledge will provide the rule more accurate to the situation. For the core reload problem, such a rule specialization could be employed to defeat or append general constraints, constraints that are inherited by a specific phase of the solution subordinate to an overall control strategy. For example, when either the strategy to distribute the less reactive fuel or rotate fuel is invoked a local constraint to avoid exchange with fuel of greater reactivity is meant to supplement general position constraints. By inheriting the general constraints to the strategy under control a way of specializing constraints is achieved.

A hybrid environment allows independent reasoning and action to be given an object instance, the reasoning associated with the object's knowledge source and the object's methods the source of its actions. An object may then operate as an expert. Experts may be created that each observe a specialized behavior of a system, recognize problems or malfunctions, then collect resources to analyze and fix the problem. The term distributed problem solving has been given to a process when several such expert objects work cooperatively on a problem. Example situations of distributed problem solving for operation of a communications satellite, factory, and electronic circuit diagnosis have been discussed elsewhere (Fikes,1985;Ramamoorthy,1988).

V.B. Spacial representation

Due to power symmetry requirements of the core, only the lower right quarter of a full sized core loading needs be modeled and instead bundles in the remaining map quadrants are matched to the solution. Bundle map position and orientation both influence the power distribution and must be represented. Therefore, a means is necessary to identify the corner of each bundle to a location in the map. Further, the bundle may at times be best represented either by reference to neighboring bundles or according to its location in the

core. The following sections identify both absolute and relative coordinate systems for bundle position and orientation in the map.

V.B.1. Map position

Bundle map position is identified by row and column from the core center of the quarter core (Figure V.1a). There at each map coordinate is represented four quadrants so the position of each bundle's corners may be treated. Two coordinate systems are used to identify position of a bundle quadrant; an absolute indexing scheme to identify the quadrant position by map row and column (Figure V.1b) and a relative indexing scheme that identifies the quadrant position to the bundle (Figure V.1c). If x@y symbolizes a point with x as the row coordinate position and y as the column coordinate position the following relations may be used to convert between coordinate systems

$$\text{absolute quadrant} = 2(\text{bundle position} - 1@1) + \text{relative quadrant}$$

$$\text{relative quadrant} = \text{mod}(\text{absolute quadrant}, 2)$$

$$\text{bundle position} = \text{absolute quadrant} / 2$$

where each parameter is a point, the third equation uses integer division, and the modulus function behaves like

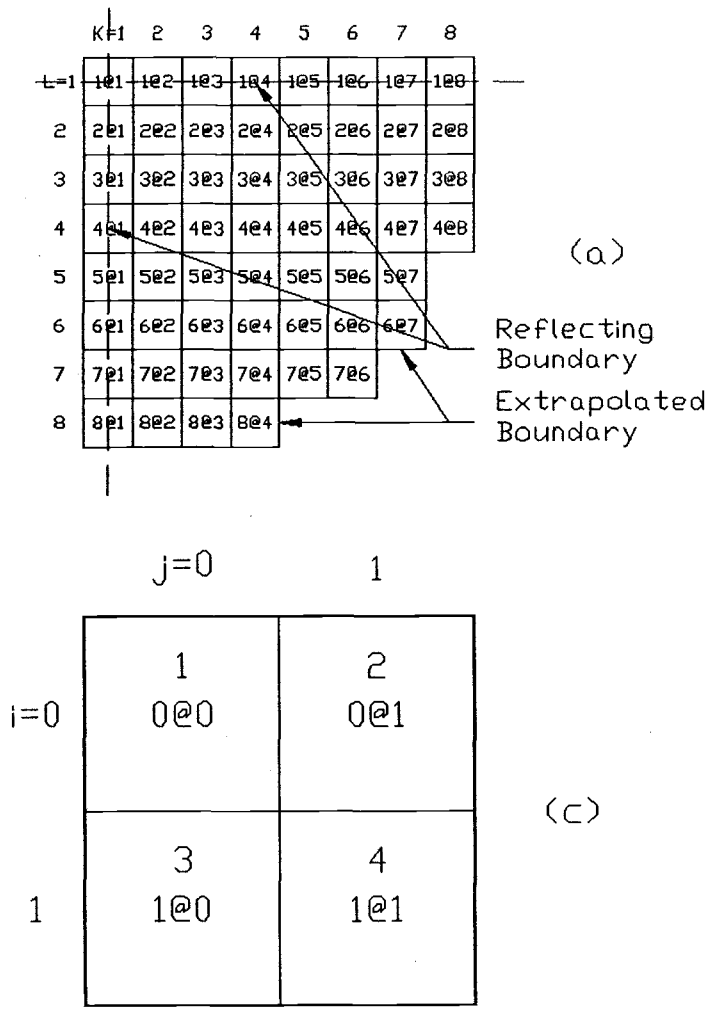

Figure V.1. Spacial representation scheme. (a) Bundle, (b) Absolute quadrant indexing, (c) Relative quadrant indexing.

the FORTRAN function of the same name.

V.B.2. Rotation

Bundle orientation in any one map position assumes one of four distinct states if the bundle cross section is square as all LWR assemblies are. Therefore it was convenient to name each of these states by an integer from one to four. Like map position, bundle orientation may be either relative or absolute. Figure V.2 attempts to illustrate representation of bundle orientation. Each bundle has corners that physically move with it lettered A, B, C, and D. When a bundle is placed in the map, its corners are identified with the quadrants of the bundle's position. A coordinate system with each quadrant numbered from one to four clockwise represents the orientation of the bundle. For example, if the bundle key (corner A) is in the upper right hand corner its orientation is one. Rotation is described as a clockwise offset from the bundle's original orientation. To translate between coordinate systems the following relations are given:

$$r_f = \mathrm{mod}(r_i - 1 + \mathrm{offset}, 4) + 1$$

$$\mathrm{offset} = \mathrm{mod}(r_f - r_i + 4, 4)$$

where r_f and r_i are the final and initial orientation (i.e., absolute rotation) of the bundle. Table V.1

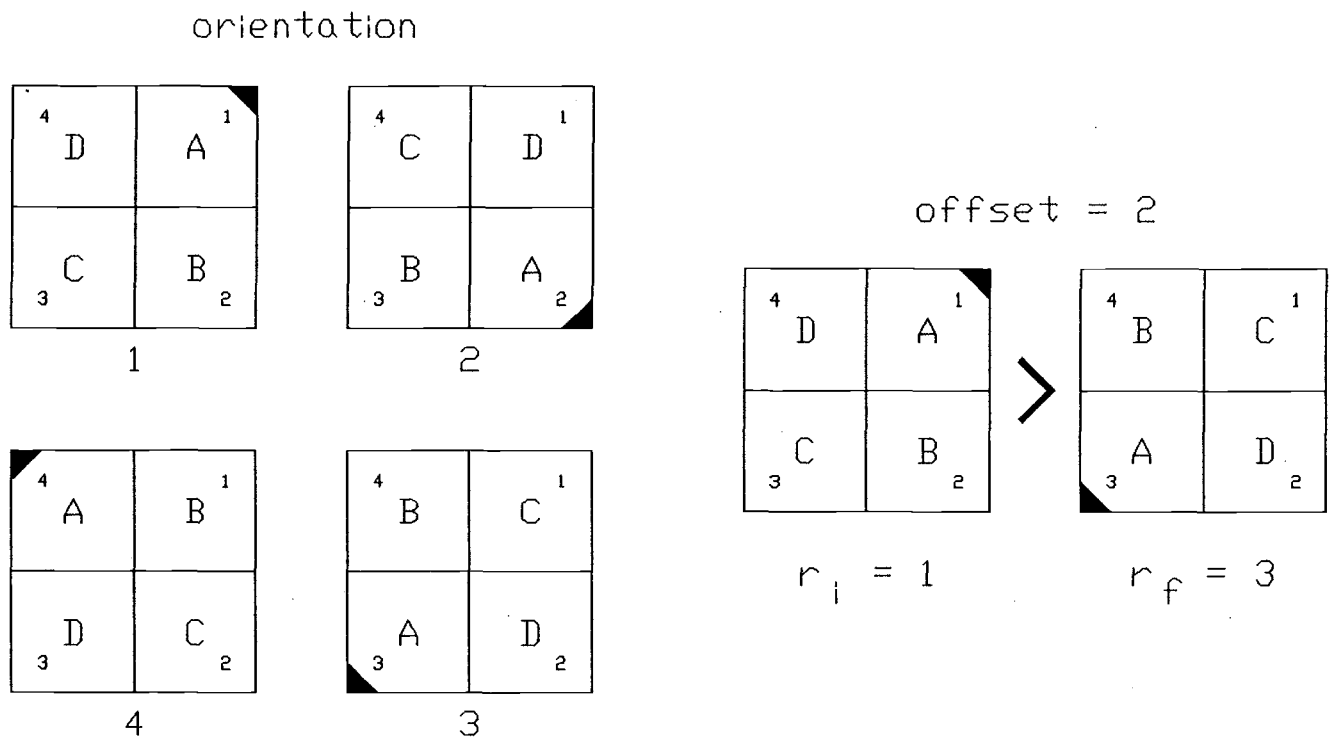

Figure V.2. Bundle orientation representation scheme.

provides an additional translation for where the relative quadrant a bundle corner is placed for a given orientation.

V.C. Summary

For the problem formulation presented earlier the hybrid environment appears the most flexible for a unified representation of objects, procedures, and rules. Two coordinate systems, one absolute to map position, and the other relative to the bundle, are used to represent bundle quadrants spatially. A third coordinate system to index bundle position in the map is necessary. Either a clockwise offset or orientation index defines the rotation of a bundle.

Since modification is necessary as the expert system is evaluated, an implementation using expert system tools that provide ease of incremental improvement is needed during development. The next section discusses the tools and appropriate support facilities to implement the core reload problem.

Table V.1. Correspondence between bundle corner and relative quadrant for a given orientation.

Orientation	Corner A	B	C	D
1	1@0	1@1	0@1	0@0
2	1@1	0@1	0@0	1@0
3	0@1	0@0	1@0	1@1
4	0@0	1@0	1@1	0@1

VI. IMPLEMENTATION

A number of expert systems have been written in the past to solve problems in diverse areas of business, science, medicine, and engineering. As expert systems are developed, valuable programming tools to aid the developer maintain and implement his application arise. So among concerns the knowledge based system developer has upon bounding the problem domain so that it is financially and computationally tractable, his selection of appropriate applications now often takes the form of relating candidate problems to known tools.

Often tools were developed for a particular application and then ways were sought to conserve time by using them when developing other expert systems. Even for tools that have been well studied, however, it is difficult to know what type of problems a given knowledge engineering tool will solve best. It is important when developing an expert system to seek a representation that models the way the expert and user thinks in rather than any suitable manner the knowledge can be coded in any one language. This simplifies understanding the system once verification and maintenance are required. A simplified understanding is, after all, one of the reasons why particular tools exist. This section discusses the appropriate problem class, the tools needed for a proper implementation,

and the features included to ease user interaction.

VI.A Problem classification

Three distinct ways for classifying problem solving methods have emerged from the extensive research performed on problem solving in AI (Simon,1983). The first class of problem solving includes search, whereby a number of alternate paths in a state space are investigated and operators are applied to traverse the space from one state to another. The search constitutes repeatedly choosing moves and evaluating states to determine a solution in an efficient manner.

Refinement is the second class of problem solving whereby the problem to be solved requires continual reformulation based on new information generated by the process. Rules are applied to deduce new conditions of the problem state from previous deductions and conditions. Solving the problem requires accumulating more and more information by inference until a conclusion can be reached.

The third class of problem solving is constraint satisfaction which considers a set of solutions to a problem then reduces that set to a unique solution or subset of solutions that satisfies all constraints. Constraint satisfaction problems do not require a particular search method of their own and may be solved

with any typical search strategy. What distinguishes this classification of problem solution is a list of changing constraints which the problem state and search direction must satisfy as the problem is solved. Many design tasks may be viewed as constraint satisfaction problems in which a design must be created within fixed limits on materials, cost, and time.

Solving the core reload problem is more of a trial and error process than an exact science; a mathematical solution for the minimum power peak loading is intractable provided any practical conditions for a loading. Among all possible configurations of a loading considered, then, constraints are applied to consider only the best of moves. The solution involves guessing and repetition of constraints and heuristics that generally remain consistent with each new loading and so should be considered a search class of problem solving. On the other hand, the task may be considered a constraint satisfaction problem particularly if the problem was to change size or restrictions of the constraints list with each loading.

VI.B Review of tools

A number of candidate tools were reviewed for implementing the core reload problem, however the majority were rule based systems similar to EMYCIN that

are strictly backward chaining and cannot allow rule iteration. A hybrid expert system environment or object oriented language with bit mapped graphics was at last recognized as the software most appropriate for this application. The three hybrid systems analyzed were Goldworks, NExpert, and KEE. NExpert, however, does not currently possess end-user interactive graphic tools that are supplied at a price less than 10 times the cost of NExpert itself. Since the review, a new version of Goldworks, Goldworks II, was released that has capabilities comparable to KEE, the high end environment in price and performance. Both KEE and Goldworks run on a 386 IBM PC based machine and require 10 megabytes of main memory.

A version of Smalltalk, an object oriented language with bit mapped graphics, was discovered that does not require a machine of considerable performance and is much less expensive than the two hybrid environment alternatives above. Its capabilities are satisfactory for a demonstration of the core reload problem.

VI.C Power calculation

A two group two dimensional diffusion calculation was required with enough detail to calculate powers for each bundle corner in a quarter core loading. The power calculation is too complex numerically to implement in

Smalltalk and so it is written for a FORTRAN compiler. Since the PC version of Smalltalk cannot execute a FORTRAN program within its environment, it was necessary to program a call to DOS where the power calculation could be performed. Information between the expert system and the FORTRAN program is passed by files. Given a suitable communications interface, however, the power calculation code could reside on a separate computer and be given the ability to cease execution and hibernate while its results are being examined by the expert system. This avoids suspending the expert system in Smalltalk each time a loading configuration is evaluated.

Appendix A presents the power calculation program. The program PWRCALC reads from a library file nuclear cross section correlations by each bundle's cross section index. All information regarding the loading to be evaluated is sent to the power calculation from Smalltalk by a file. This includes each bundle name, cross section index, position, and exposure by corner, information on the soluble boron concentration, the current loading exposure, and the objective exposure. In exchange, PWRCALC updates the database entries for the new exposure and adds the bundle power and reactivity by quadrant, the loading effective multiplication factor, and the position of the maximum

power bundle then writes this to an output file.

If the expert system were to represent a BWR it could not treat the axial power distribution with PWRCALC since this varies by height. If PWRCALC were used, the loading optimization would be based on a 2-D calculation that is felt to be approximate, however the loading results could be checked at intervals by more precise but time consuming 3-D evaluations. The practice of intermittently verifying the power calculation using a more exact method has found use in optimal loading searches in the past (Jonsson,1986).

VI.D User interaction features

Smalltalk's programming facilities for interactive windows and bit mapped graphics enabled development of an expert system with full workstation capabilities. The user may use a mouse to interactively monitor and correct solution progress or display and edit the loading configuration. Sections of the map also may be selected upon reactivity or power threshold. The multiwindowed environment of Smalltalk allows the user to create and select windows in order to examine loadings generated or choose between multiple Shuffle processes operating independently.

There are three characteristic panes of Shuffle's windows: a text pane from which text or data may be

displayed or modified; a list pane that offers the user a selection of loadings or moves to choose from; and a load pane that displays the loading configuration for manipulation. The user cannot directly modify a loading in the load pane since this would change its historical significance in the solution, but instead moves may be generated which when implemented create a new loading. The contents of the load pane and text pane may be printed for documentation. Finally, the text pane may answer simple queries from the user about the status of some parameters in the solution or perform some action.

A number of bundle selection and placement aids are available to the user to support his common sense knowledge in search of an optimal core configuration. These tools can specialize the most fundamental operations to select, sort, and position bundles any fuel management expert is familiar with. The aids are implemented in the load pane and include the capability to

1. Apply rules for symmetric bundle placement about the map centerlines as necessary when the user exchanges two bundles.

2. Access and display records of the pool databases.

3. Allow the user to select criteria to accept or reject bundles on the basis of a threshold or range of power, reactivity, or exposure. When filling an empty map location, the user may select bundles in a pool from a collection sorted to match the criteria of the vacancy.

4. Identify to the user by shade or color bundles in the map that have a magnitude of reactivity, exposure, or power within some specified range.

5. An option to prohibit manual bundle placement by the user if it violates position constraints and offer an explanation.

6. Moves, when generated manually, may comprise any number of exchanges, rotations, or interchanges with pools.

7. Retract or undo any number of steps comprising a move.

Figure VI.1 shows Shuffle's windows and their features. These are:

Shuffle executive window -- contains a text pane to display status of the solution and a list pane to select one of any loadings generated for display in its load pane. This window controls the search, opens supplementary windows, or trims the search space of loadings upon command.

Loading edit window -- the user may display and edit the configuration thereby generating moves that are added to its moves-to-apply list and displayed in the move list pane. A text pane displays error messages encountered during editing. Any move may be selected and implemented directly.

Tree window -- displays the state space search as a tree from which the user may print or select loadings for inspection in the Shuffle executive window.

Explain window -- a reasoning trace of rules as they execute are displayed here and the explanation capability may be toggled on or off.

Plot window -- displays a plot of the progress of the solution such as loading effective multiplication versus the peak power factor.

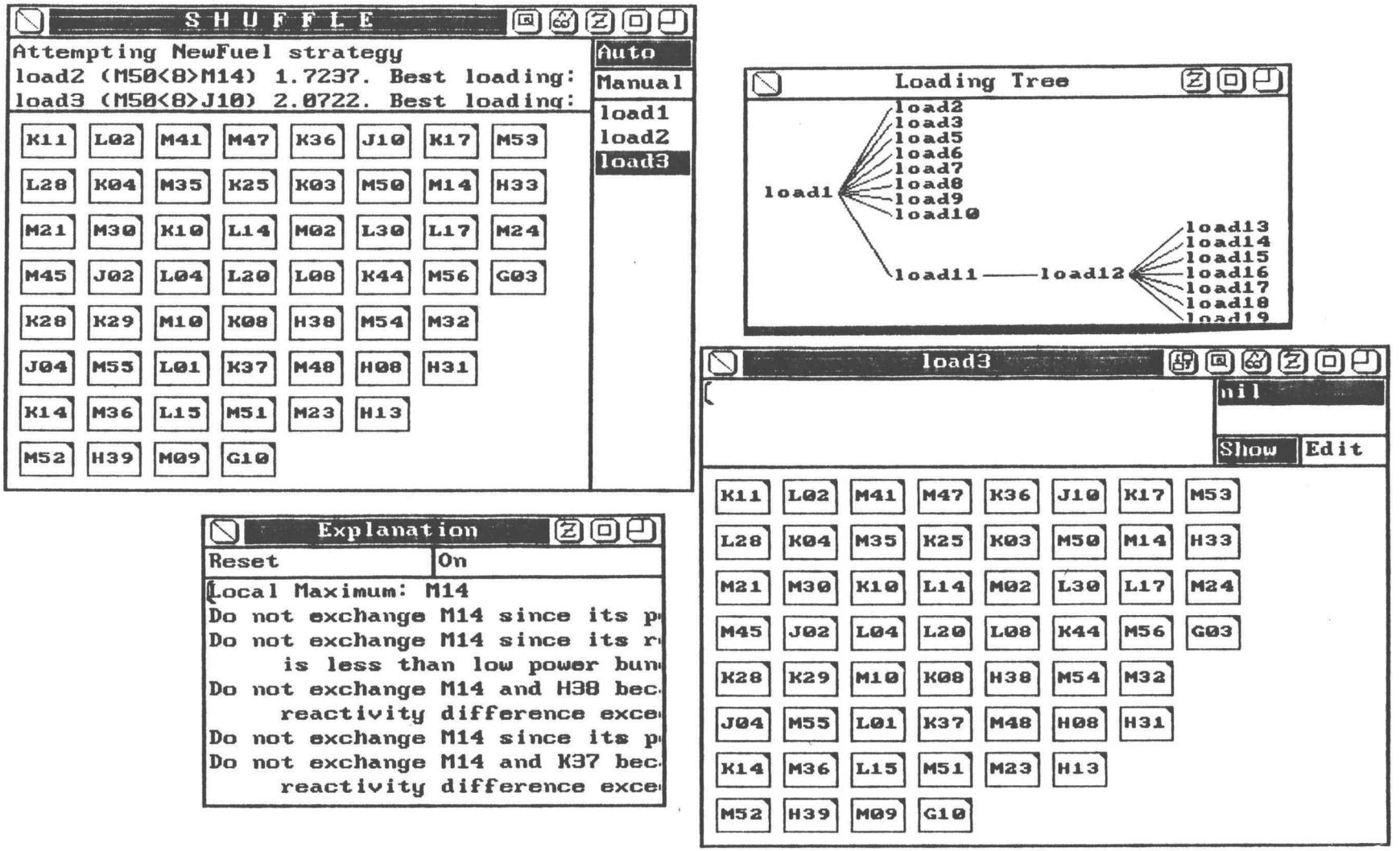

Figure VI.1. Interactive windows of the Shuffle expert system.

VI.E System breakdown

Several different modules or objects were developed for the Shuffle expert system either to represent the structure of the problem, to implement search, or facilitate an interactive environment. A functional breakdown of the major objects of the Shuffle expert system is shown in Figure VI.2, many of which associate with windows already discussed. The system at its simplest level consists of a single instance of a Shuffle executive, a loading database, and a search module. Instances of load editor, plot, or tree objects are created when they are needed by the user and are controlled by the Shuffle executive.

VI.E.1 Loading database

The solution of Shuffle is recorded in an object-oriented database. The database simply has a list of trial loadings that grows as the solution proceeds. Each loading not only records a map configuration of bundles that retain all attributes of power, exposure, reactivity, etc. but records the move that generated the loading, all the moves applied to the loading to generate new loadings, and all moves yet to be applied to the loading. Finally, each loading stores a record of the search strategy that generated it. One loading

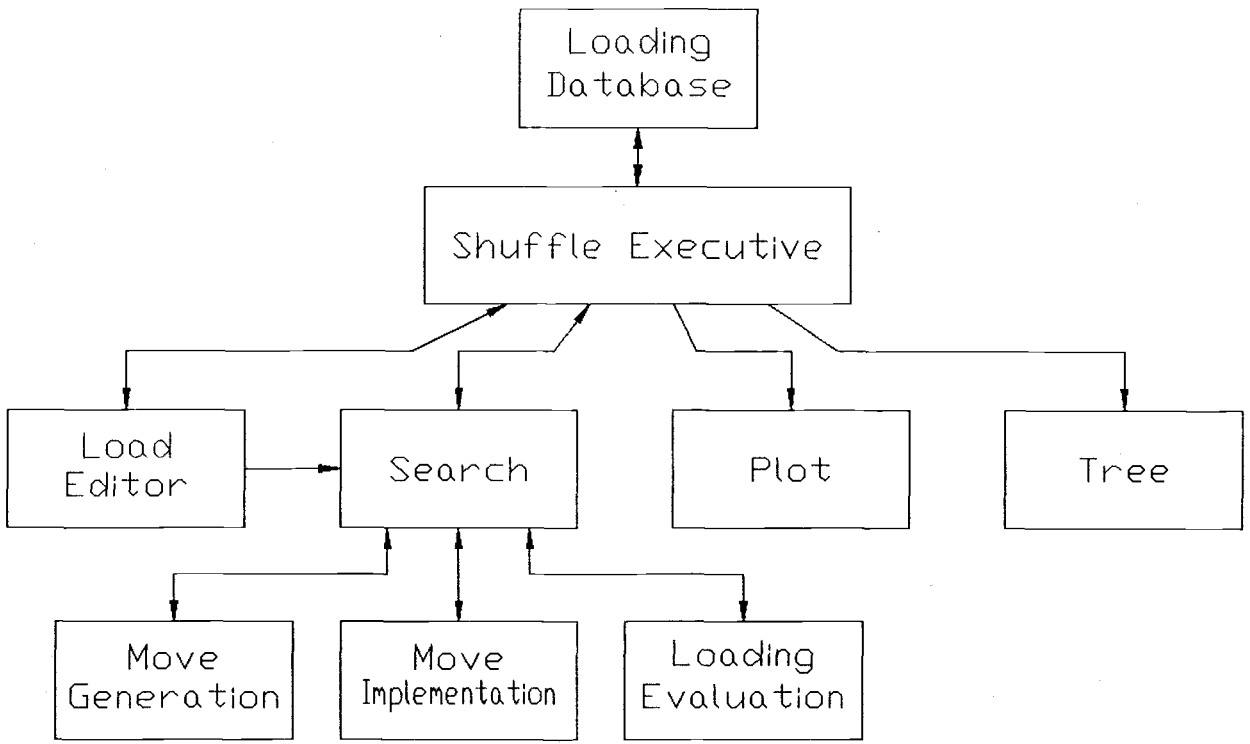

Figure VI.2. Operational breakdown of Shuffle.

therefore has all the information of the state space required to backtrack and reinvoke the search from the time of its creation. The compact yet extensive nature of this database demonstrates the power of object-oriented representation.

VI.E.2 Search

The search object uses heuristic knowledge and a weak search method to generate, select, and implement moves to a loading then evaluate the new configuration and, if the search objective is not satisfied, choose the next loading for investigation from among all states created. Upon creation, the search object is assigned a list of strategies to be tried in succession during the solution and an objective function that, when applied to a loading, the search must minimize. Once a limit has been reached such that the current strategy exhibits diminishing progress, the search has the facility to improve circumstances by changing all this for a new strategy. It is also simple to change the weak search method even during solution. Whatever strategy is in use, the search module chooses the next move to investigate and which loading is best independent of the Shuffle executive operation.

As shown in Figure VI.2, search has three minor modules that cooperate to perform its task. The first,

move generation, uses the heuristics and constraints coded to the strategy to generate a list of moves to be kept with the loading under investigation. Move generation, however, is only performed if the move-to-apply list of the loading under investigation lacks remaining moves. The most appropriate move is selected from the list, implemented to create a new loading, and its power calculation is performed by the move implementation module. The loading evaluation module then determines if the new loading satisfies the search objective function and the next loading to be investigated is decided. The weak search algorithm resides in the loading evaluation module and additional heuristics may be used to help it direct the search under some circumstances.

The search may be initiated either by a request from the executive or load editor. If a load editor requests the search, this is a choice the user has made to implement a particular move so both loading and move must be identified to search. In either case, at completion search provides the executive with the new configuration generated to update the database and the next loading to be investigated.

VI.E.3 Shuffle executive

Instrumental to the system is the Shuffle executive

that controls the access to the loading database so that the solution state is consistent between objects. The executive controls the solution so that it proceeds in a cyclic fashion originating with the loading provided search and continuing with the next loading proposed by search. At the conclusion of the cycle, the loading just generated is added to the database and a decision is made for which loading to continue the next cycle. A cycle that extends outside the search module allows the user the option to interrupt the normal process of the search and direct it from a new state.

VI.E.4 Knowledge base

A strategy as applied in the search module is an instance of a class of hierarchically defined heuristics and constraints that provide the intelligence of the search. Figure VI.3 is a hierarchical diagram of the strategies in the current configuration of Shuffle. Note those constraints that remain consistent throughout all strategies are grouped in a superclass for common access. Smalltalk lacks a formal mechanism such as an inference engine to select and activate rules and lacks a formal rule representation. Instead Shuffle relies on the features of object-oriented programming for rule selection. Rules are programmed in the desired order and scanned

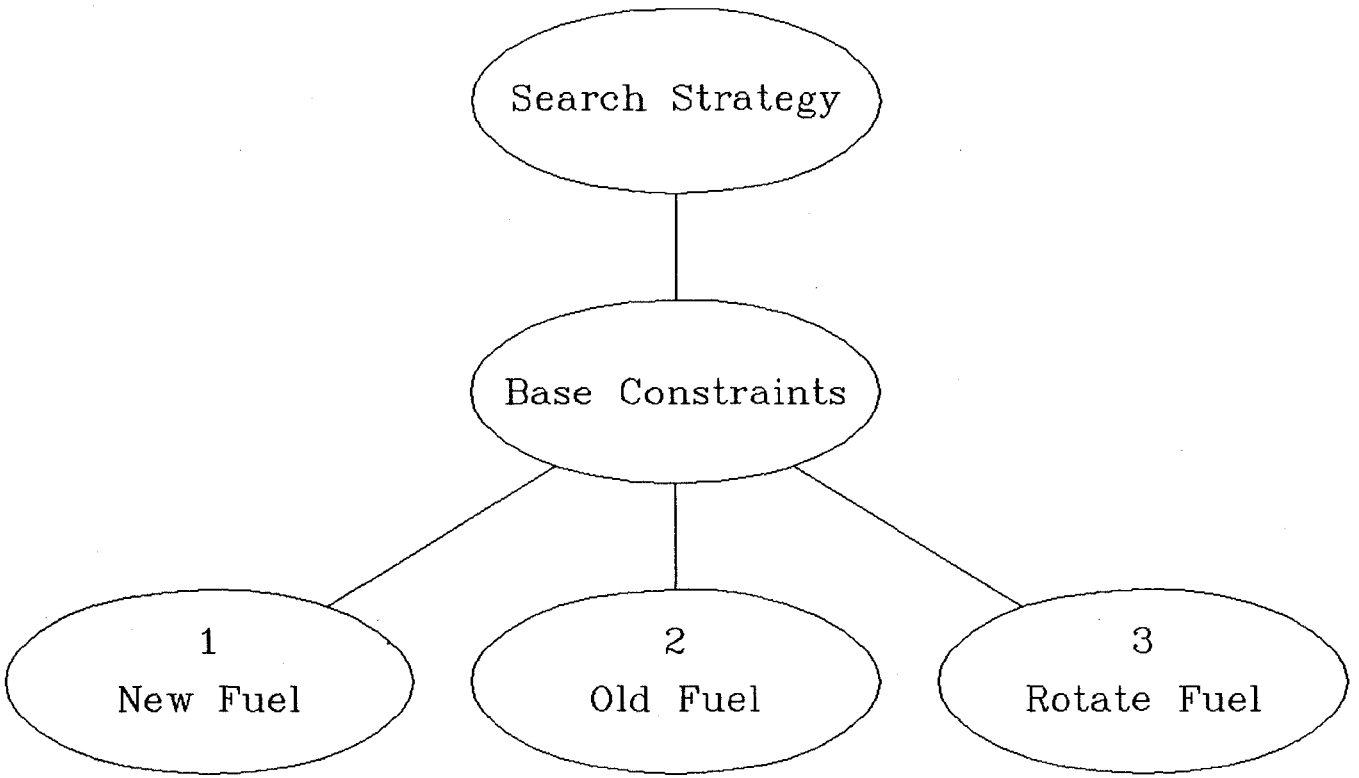

Figure VI.3. Hierarchy of Shuffle strategies.

linearly until one is found that matches. An error is displayed if no match occurs. Rules may be coded to enable a reasoning trace to be accumulated as the solution proceeds.

Regions are represented in the loading to distribute fuel according the type of loading pattern desired (e.g., out-in). The regions identified by Stout in his original Shuffle are used, however the user may name or place the regions however he chooses (Stout,1973). Figure VI.4 shows the characteristic regions of the quarter core loading as implemented which are identified as even, odd, intermediate, and periphery.

VI.F Comprehensive error checking

Comprehensive error monitoring must be introduced to detect incomplete or inconsistent data or warn against behavior that may lead to an uncertain outcome and proceed undetected through the solution. Additional coding is included to further reduce the likelihood of errors. Checking for the reasonability of a variable's numeric range, for undefined or lost elements during database operations, or for a match failure in conditions of an important rule set or block of code are three examples of such error checking implemented.

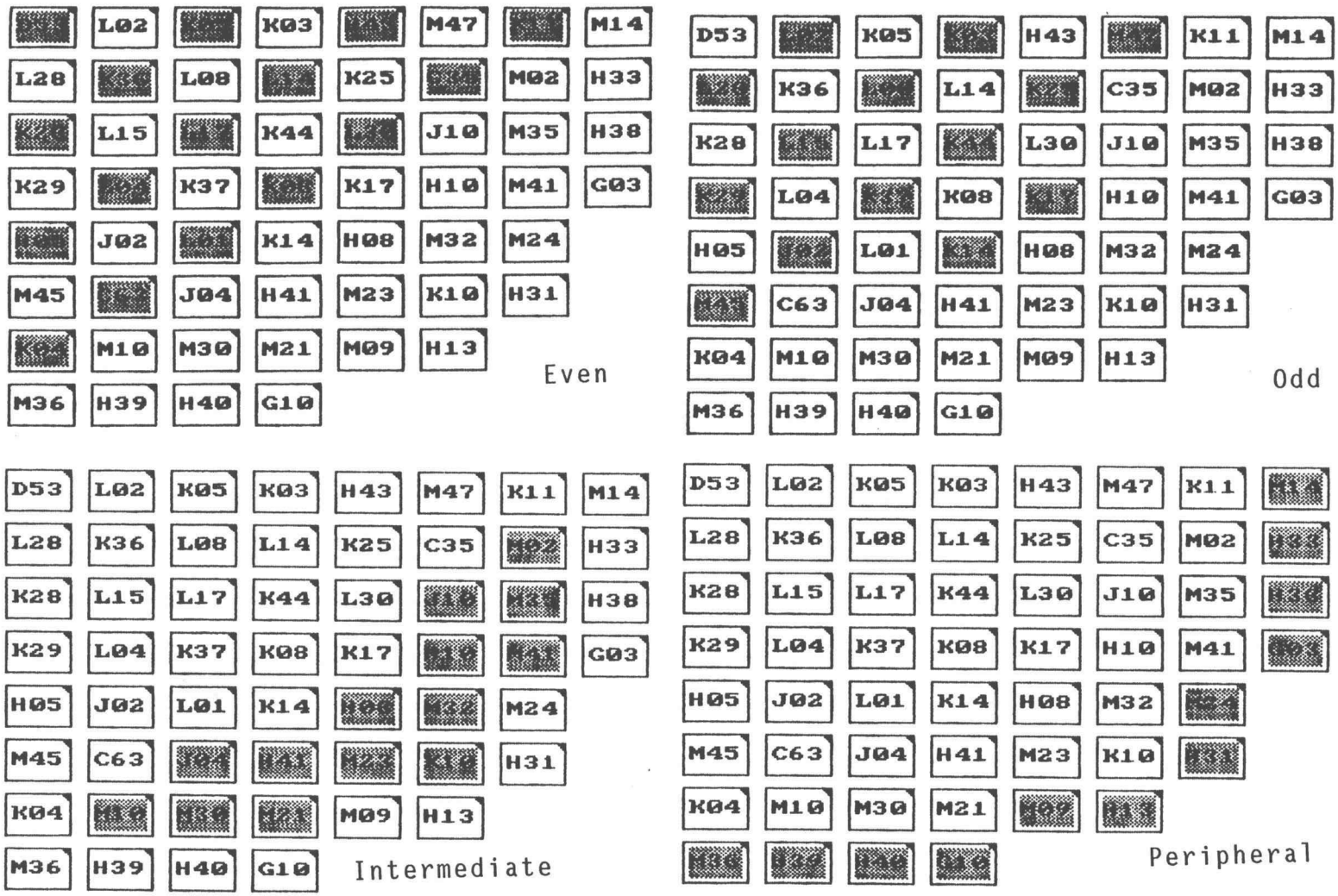

Figure VI.4. Characteristic regions for a modified out-in pattern.

VII.G Explanation facility

Expert systems, unlike knowledge based systems, possess the capability to explain their reasoning upon demand. To facilitate an explanation capability in Smalltalk, new ifTrue and ifFalse methods were added which enter messages to a global parameter called Explanation. These methods, ifTrue:explainTrue: and ifFalse:explainFalse:, follow a block of code provided to generate the explanation message from evaluating expressions, parameter values, or canned text. If the explanation facility is enabled, as rules fire in Smalltalk the messages from each rule are collected in a chronological order in Explanation. The explanation facility displays the messages in the Explain window to the user. The user may reinitialize the explanation gathering process at any time or choose to disable the process completely.

The depth of explanation depends on how many rules coded with an explanation message fire and the detail of the message coded. If more information upon the process of the solution is needed to trace execution and variables, coding could be added to place messages in Explanation at key locations.

VII. EVALUATION

The evaluation process of an expert system is a continual one that should begin with the system design, extend through the early stages of development, and become increasingly formal as a developing system approaches a real-world implementation. The Shuffle expert system has not seen development to the stage which allows wide distribution and feedback has not been obtained from an expert or from users but suitability of the system has been compared to features of fuel management tools which have gained acceptance. This section discusses the major concerns for evaluation of Shuffle at this development, namely the verification of knowledge base coding and the performance of its intelligence.

VII.A Verification

During knowledge base development, a variety of errors can arise which lead to inconsistencies or gaps in the knowledge base. Verification is the process of testing and refining the system's knowledge in order to correct the errors that occur.

Features which simplify the task of ascertaining whether the knowledge base is correct and complete are available in the Shuffle expert system and the Smalltalk

environment. First, an interactive method in Shuffle allows position constraints of any chosen strategy to be tested using the loading edit window and an interface to the system's explanation facility. When bundles in the loading editor are selected interactively for exchange, rotation, or interchange with a pool and such moves are forbidden by constraints of the active strategy, an explanation of why the action is prohibited appears in the loading edit text pane and the move is not implemented. This manner allows the user to verify immediately the constraints used in each strategy. Second, an explanation trace of rules may be enabled with the explanation facility to allow a record of the conclusions of each rule to be examined during the search. Finally, the break and trace mode available in the Smalltalk environment allows the user to stop and follow progress of the solution and the rules implemented. These three verification methods may be used to investigate in progressive detail the performance of the solution.

VII.B Evaluation

Even though considerable effort is spent upon knowledge base verification, users of Shuffle may not be generally concerned whether its final recommended pattern has been reached in a "correct" way so long as

the pattern developed is appropriate. Given enough time, the base search procedure of Shuffle could survive with a poor knowledge base and arrive at a reasonable configuration eventually. Since expert systems normally are developed for those domains in which decisions are highly judgmental, the mechanisms for deciding whether the system's result was derived efficiently is difficult to define or defend. Declaring a knowledge base is more intelligent than another therefore requires an extensive and careful evaluation of the solution progress for both knowledge bases being compared. Evaluation also relies upon correct coding of the knowledge base to test it fairly. Unfortunately, a correct knowledge base may be difficult to prove.

The intelligence of the Shuffle loading search could be measured by its rate of convergence, its response to expert patterns, and the speed to correct pattern placement. Although random pattern redistribution would not seem as intelligence it is also important in terms of seeking new patterns. The following sections test the intelligence of Shuffle's knowledge base by each criteria. Note cross section correlations by exposure for each test case appear in Appendix A.

VII.B.1 Rate of convergence

Given an initial loading pattern which by no means may be a configuration by an expert, the strategies employed as well as the initial pattern determine how quickly the solution may converge to a near optimal pattern. The speed with which a near optimal pattern is approached is the subject of this first test.

A search was performed on a modified out-in pattern to investigate convergence. Figure VII.1 shows the initial pattern shaded to show power and reactivity. Typically, when the old fuel strategy is applied under 40 loadings are created before the strategy exhausts its means to generate new moves and the next strategy begins. This particular pattern is unusual since a search spawns over 100 different successor patterns and more may be generated if the user does not force the strategy to stop. When the search is performed on a Compaq Deskpro 386 operating at 25Mhz, the 100 loading patterns are completed in a little over one hour and a total of 6.2Mbytes of random access memory is required to retain all patterns.

The results are shown in Figures VII.2 and VII.3. Figure VII.2 shows the convergence upon the minimum peak power and the best loading peak power trial by trial. Figure VII.3 shows the distribution of peak loading power by loading k-effective valid for beginning of

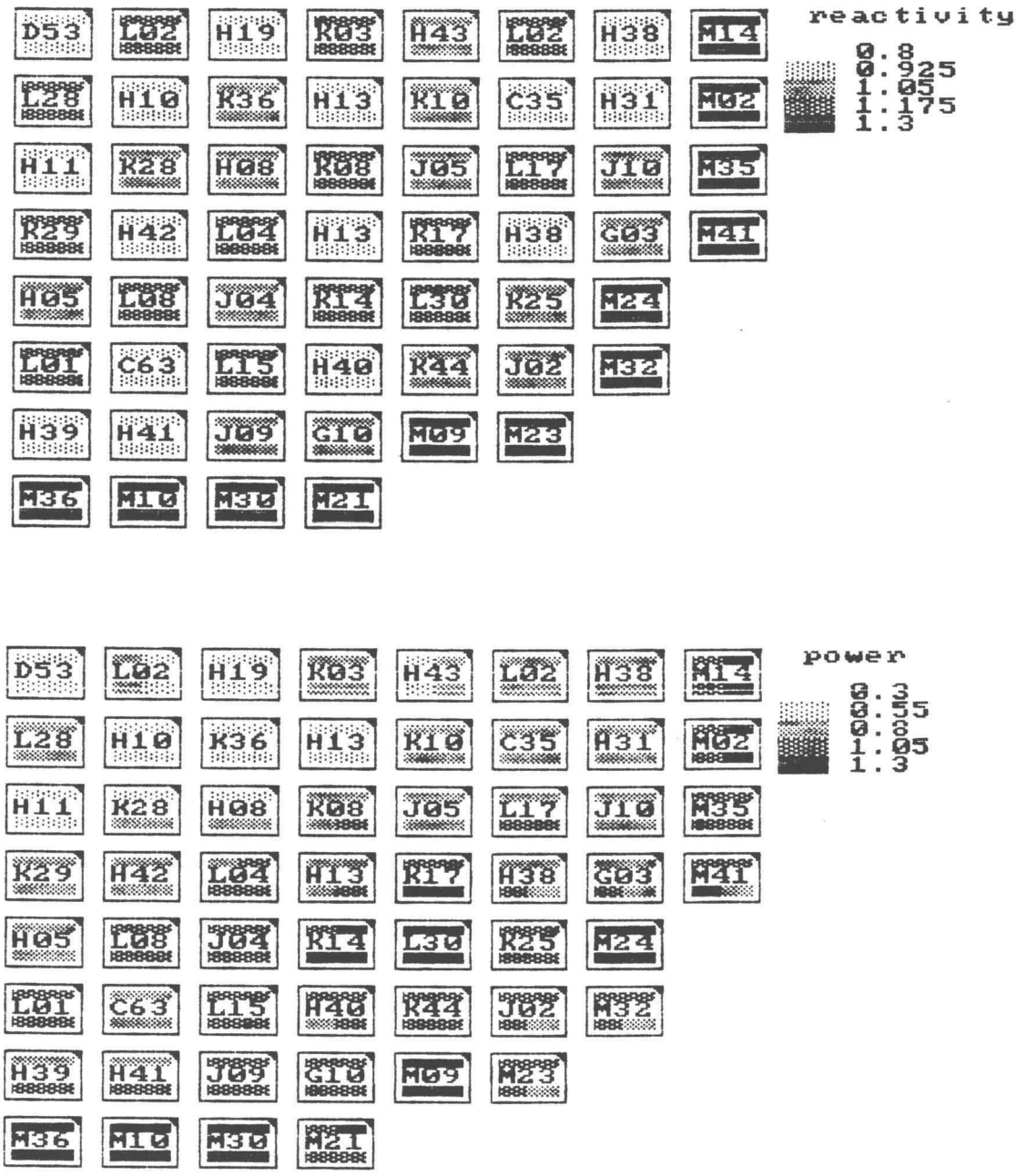

Figure VII.1. Initial pattern for tests one and two with power peak 1.5926.

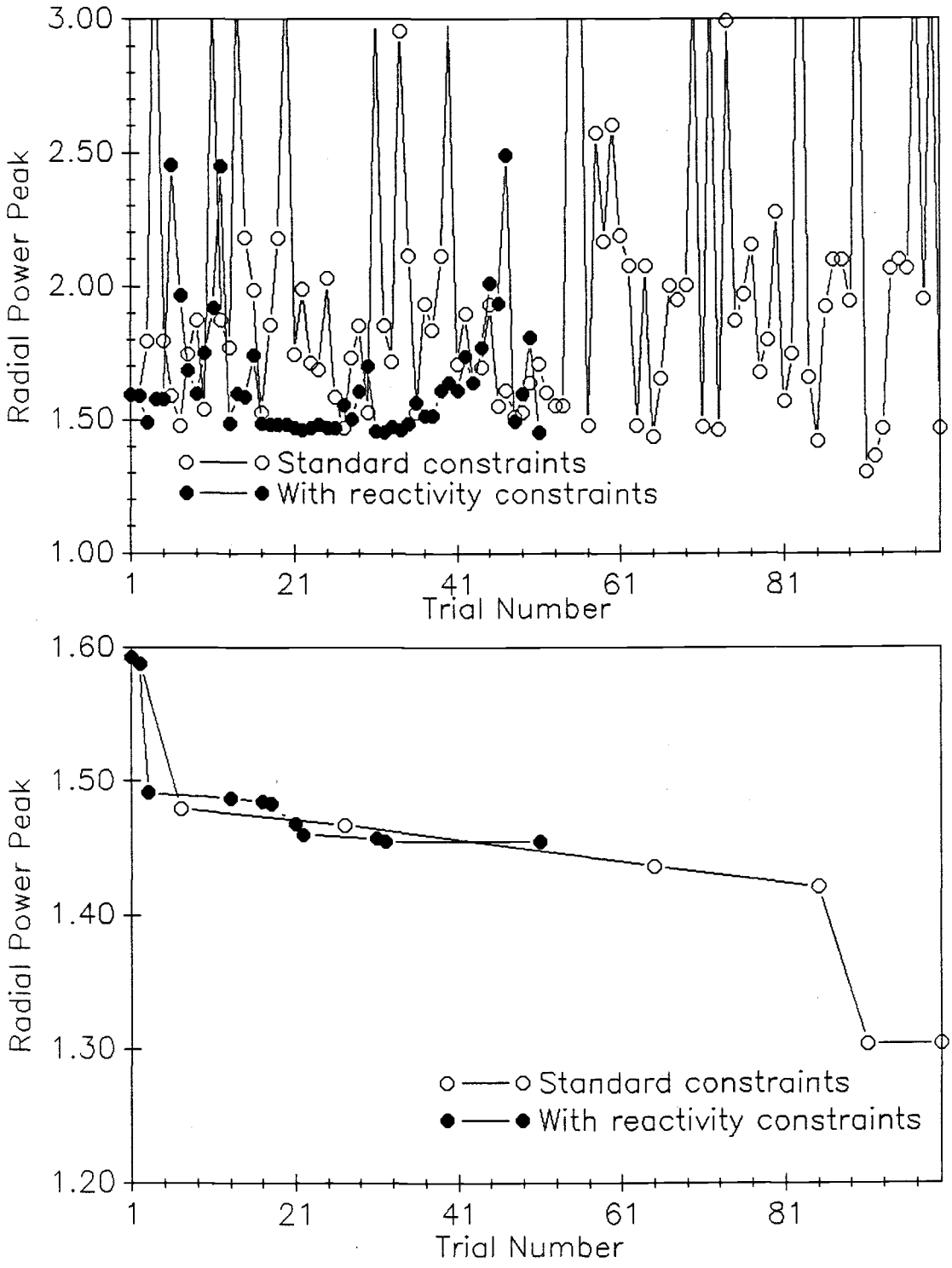

Figure VII.2. Convergence upon power peak and minimum power peak by trial for tests one and two.

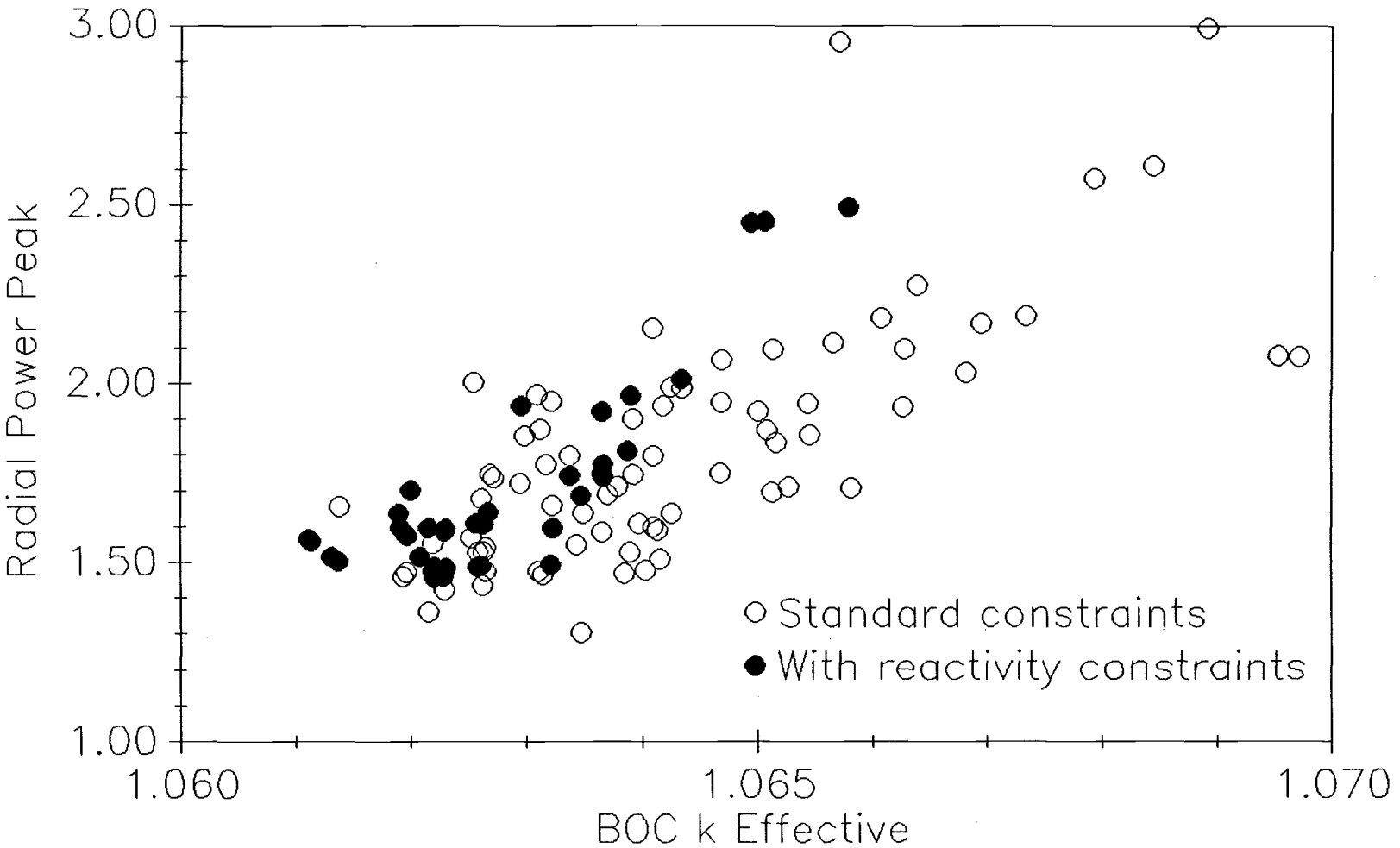

Figure VII.3. Distribution of peak power by loading effective multiplication for tests one and two.

cycle, equilibrium xenon, and no boron. Note since the radial power peak and loading k-effective tend to increase when high reactivity fuel is placed near the core center a relatively linear relationship is shown when the two parameters are plotted. The best loadings appear in the lower right of this figure since they satisfy both extended cycle length and minimal power peaking. Figure VII.2 shows the convergence to the minimum peak power loading begins rapidly at first but limited improvement is encountered as the search continues. The optimum loading of 100 trials is loading 91 with a radial power peak of 1.3035 which compares to the initial power peak of 1.5926. Loading pattern 91 is shown in Figure VII.4.

After performing this test it was realized most exchanges were poor since high reactivity bundles were being exchanged to a region already carefully balanced by a low reactivity bundle surrounded by high reactivity bundles. Sometimes a power peak in excess of 3 was obtained. To avoid cases where a move may be implemented which could introduce a great disturbance in power balance, a constraint was added to the old fuel strategy which rejected exchanges between bundles with a reactivity difference of 0.18. The value 0.18 is arbitrary and may be adjusted to filter out more or fewer exchanges. A second test case was run on the same

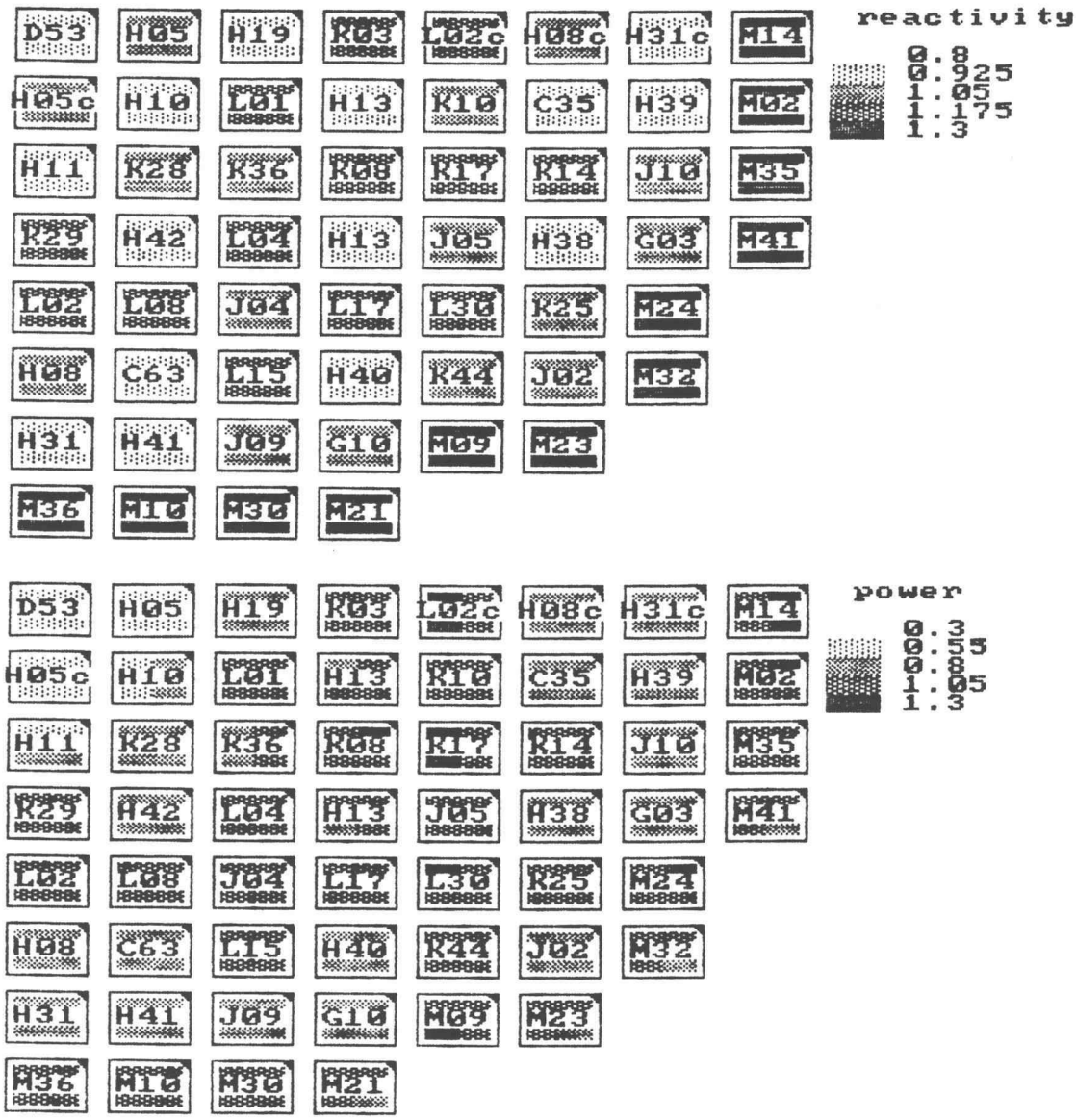

Figure VII.4. Optimal loading pattern 91 for test one with power peak 1.3035.

loading, a much faster convergence was observed, and only 32 loadings were generated before the strategy failed. However, the same constraint that increased convergence by eliminating most of the poor moves eliminated chances to generate a loading equivalent to loading 91 of test one. Instead a loading with a power peak of 1.4553 was obtained (Figure VII.5). It can be concluded that convergence of the solution may be accelerated sometimes at the risk of losing the capability to explore for better patterns.

VII.B.2 Response to expert pattern

If the shuffling search begins with a loading pattern configured carefully by an expert, the expert system must not conclude this is an inferior pattern unless the expert used for his design some constraints unknown by the expert system. Instead it should be more likely an exhaustive search will not discover an improved pattern.

VII.B.3 Rapid placement correction

Often the difference between a loading pattern proposed by an expert and an unacceptable pattern is only a single exchange. This test determines how long the expert system takes to correct an expert pattern

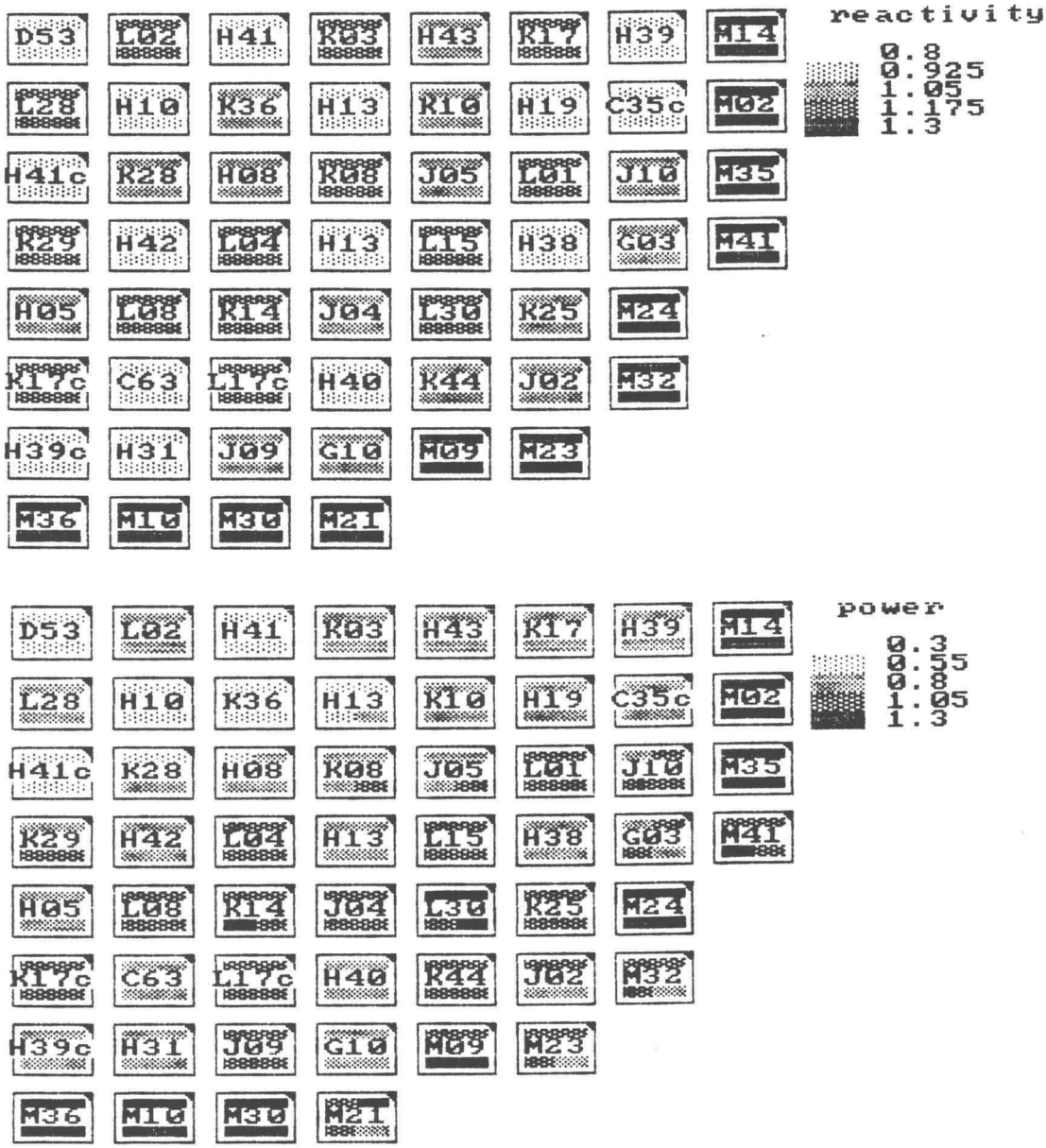

Figure VII.5. Optimal loading pattern 32 for test two with power peak 1.4553.

when it is disturbed by a single move. Experience with several initial loadings that differ by only one exchange showed many times that the search implements moves which stray away from the expert configuration altogether. Fortunately, a comparable pattern is eventually found.

The best loading pattern of test one was chosen as reference, bundles L17 and J04 were exchanged, and a search was initiated. As anticipated, the system did not determine an exchange of L14 and J04 would revert to the original pattern but continued applying new moves. Suprisingly, after the seventh move an improved loading was obtained with a power peak of 1.2969. Figures VII.6 to VII.9 show the intial pattern, convergence, and final pattern of test three.

VII.B.4 Redistribution of patterns

This test is similar to the placement correction test above but specifically evaluates the performance of the expert system for creating loading patterns which are markedly different in configuration but similar in satisfying constraints of power peak value. A graph such as Figure VII.3 helps visualize where patterns which satisfy similar constraints are grouped and helps the user locate alternative patterns.

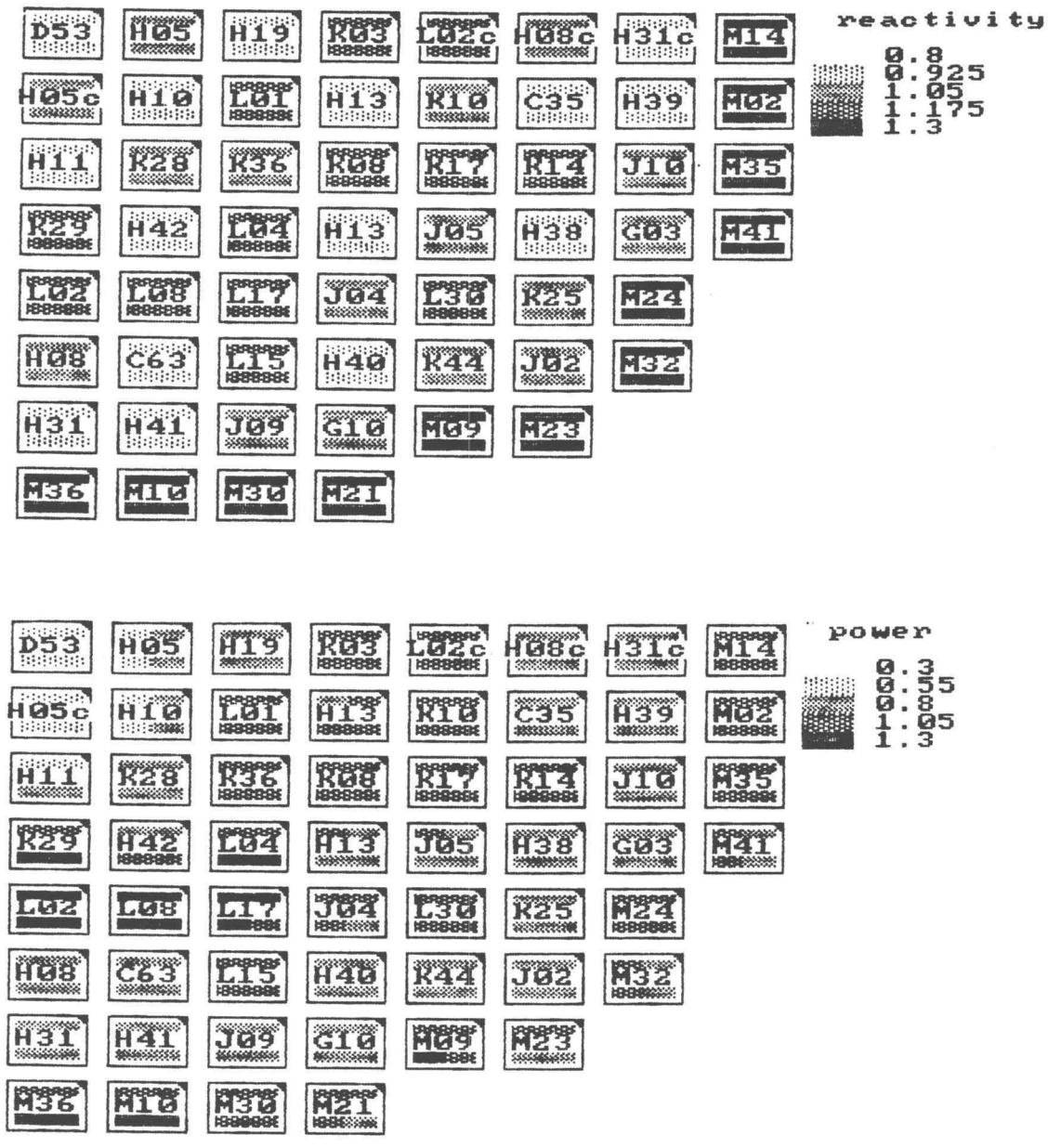

Figure VII.6. Initial pattern for test three with power peak 1.4541 as derived from loading pattern 91 of test one with exchange of L17 and J04.

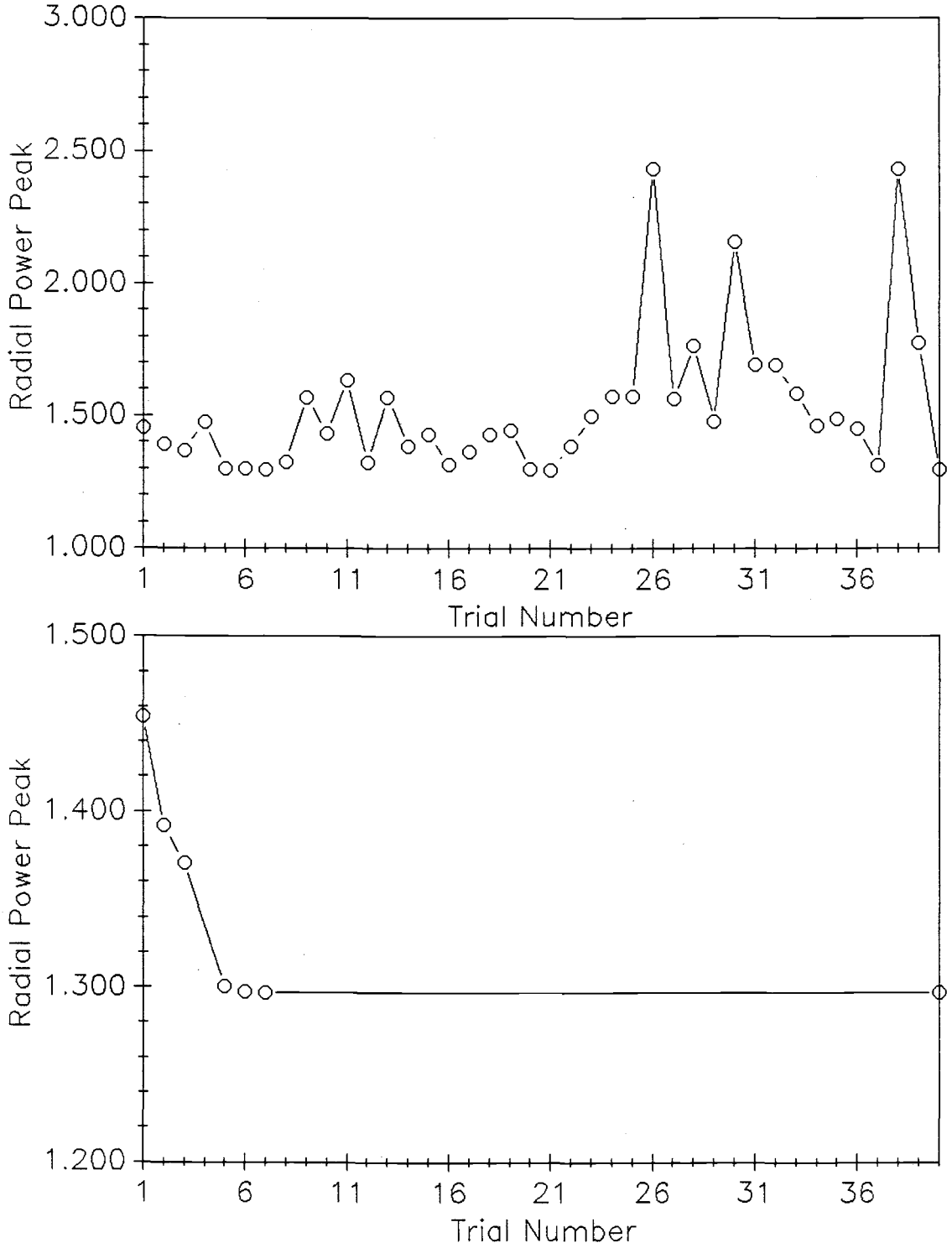

Figure VII.7. Convergence upon power peak and minimum power peak by trial for test three.

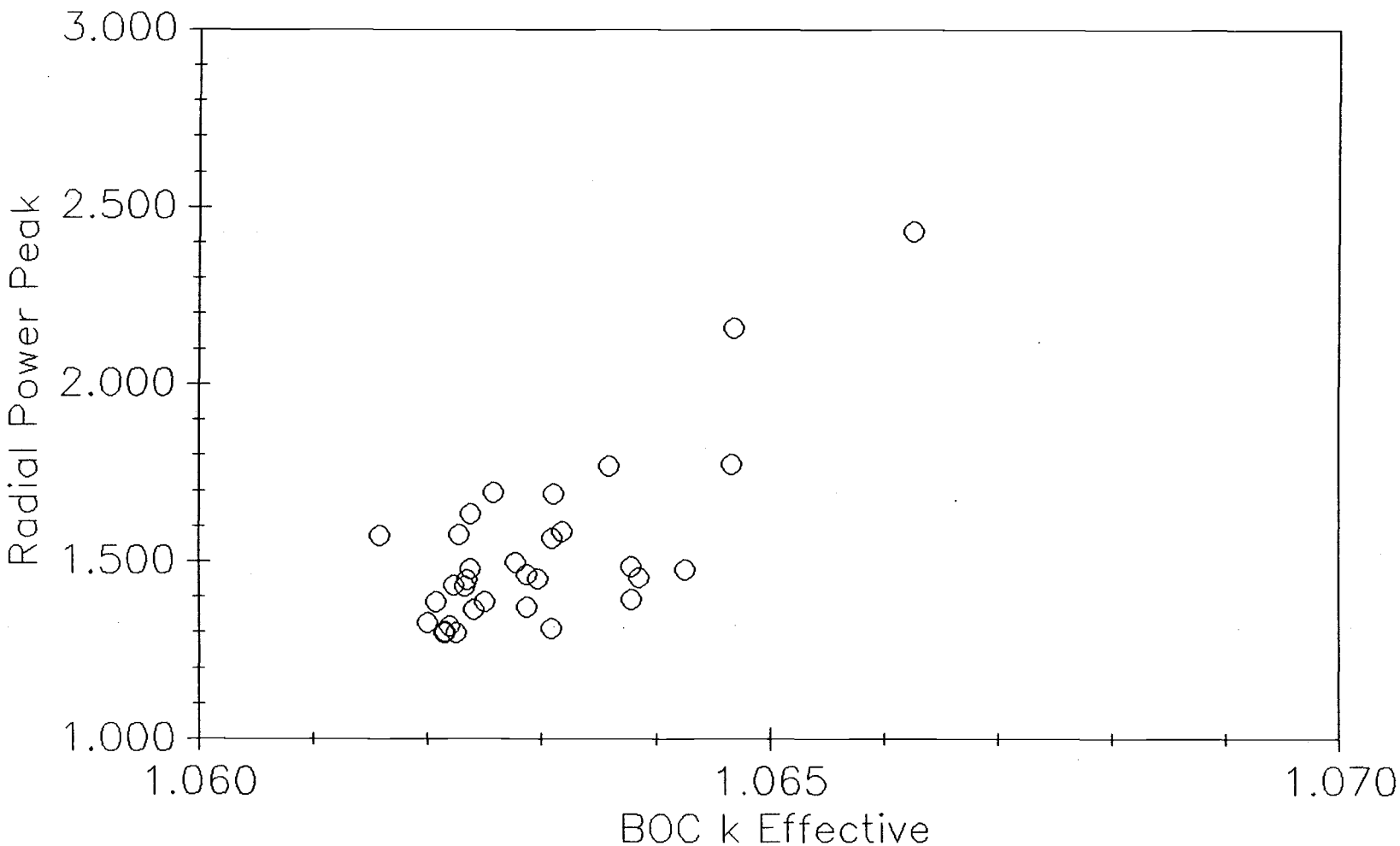

Figure VII.8. Distribution of peak power by loading effective multiplication for test three.

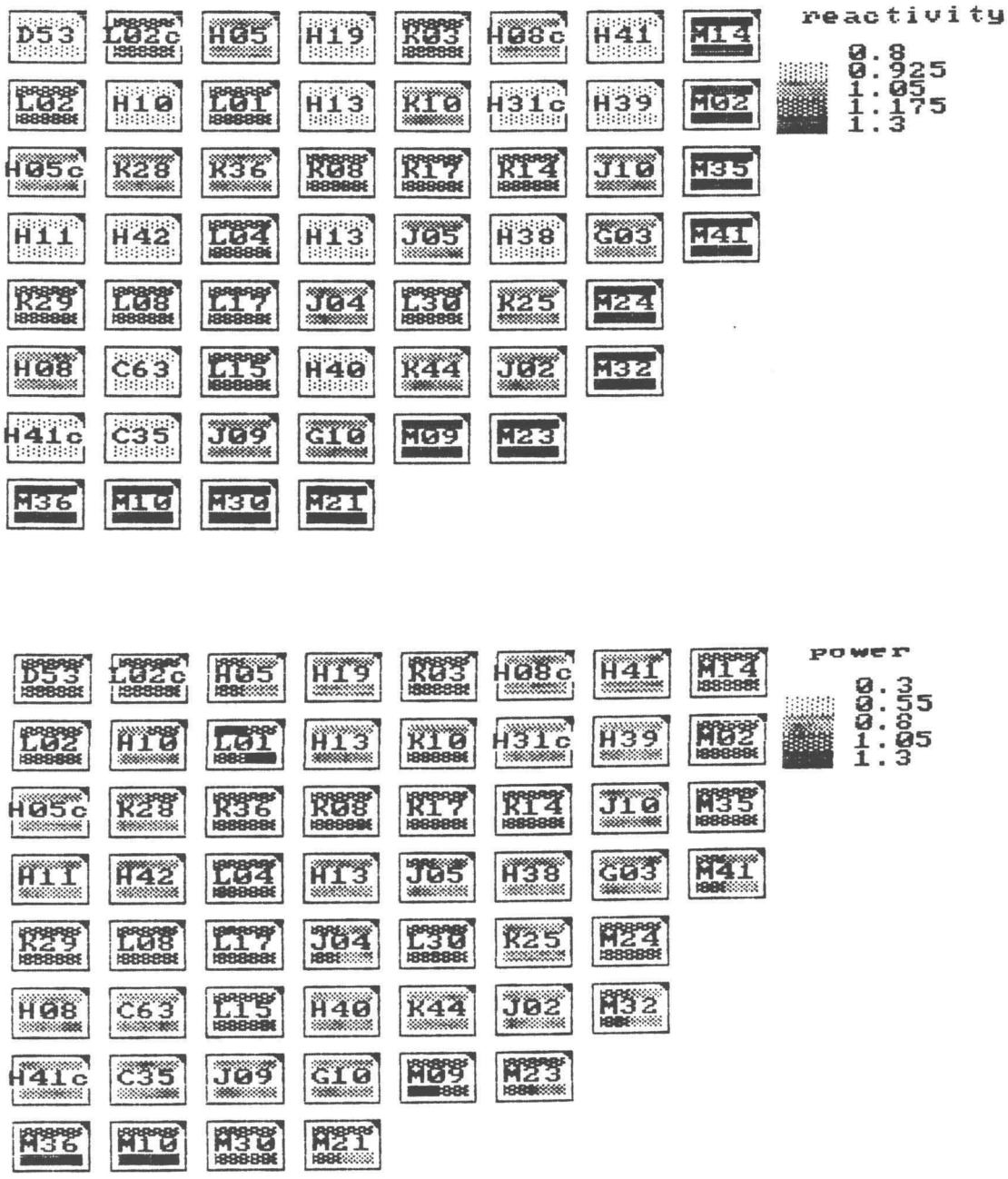

Figure VII.9. Optimal loading pattern 7 for test three with power peak 1.2969.

VII.C New strategies for investigation

After observing the effectiveness of moves generated by the expert system a number of limitations were recognized in the search. Based on observations, the background of research performed on loading configuration, and an understanding of the search process, some recommendations may be made upon investigating alternate strategies.

VII.C.1 Detailed knowledge on placement following exchange

When an exchange is evaluated one of the primary constraints applied verifies that high reactivity fuel is not placed in a region of high power. The exchange may meet this criteria in the region near the local power peak but when the exchange is performed the bundle removed from this region often replaces a lower reactivity bundle in a location surrounded by high reactivity bundles. The power peak often shifts to this new location and a greater power imbalance results. It is realized that the consequences of such a move can at best be estimated a priori and eliminating all suspect moves would eliminate many good patterns. A reactivity threshold between the bundle exchanged and the bundles surrounding its proposed location could be used to determine when such moves are used. Similarly, Galerpin

et. al. avoided a like problem in placement by eliminating moves which would place fresh fuel adjacent to once burned fuel (Galerpin,1989).

VII.C.2 One eighth symmetry requirements

When high reactivity fuel is being distributed a single move can impose a power imbalance on one side of the one eighth symmetry line which could be corrected by performing a similar move in the transpose position. No rules exist in the current knowledge base, however, which recognize the power offset and attempt the balancing move. Stout recommended that one quarter core symmetry be observed reasoning that one eighth core unnecessarily limited the possible loading patterns (Stout,1972). However, some improvement in convergence could be obtained if one eighth symmetry is used initially in the search. Test three, for example, used a one eighth symmetry constraint for moves of the new fuel strategy and obtained a 1.2926 power peak rather than 1.3240 when the conventional one quarter symmetry is used.

VII.C.3 Search method

The branching factor of a node in a state space graph is given by the number of arcs (i.e. moves) which

may be applied from the node. In the strategies reviewed here, the old fuel strategy has the largest branching factor since it is common to have 10 or more moves available to be applied to a single state (compare this to the branching factor of at most two for the new fuel strategy). Because of the large number of branches it may be feasible to consider a different weak search method such as best first search or branch and bound so that the number of moves generated but never implemented for each loading could be better utilized in the search.

VIII. ENHANCEMENTS

Once an initial implementation for solving the core reload problem has been accomplished, a number of extensions can enhance the aid and further simplify solutions to related areas of fuel management.

VIII.A Low leakage fuel management

Fuel management calculations are generally solved in a forward direction, such that the flux and power distribution is solved from a given loading configuration at BOC toward EOC. If the constraints are not satisfied, the core loading must be changed and the calculations are performed forward again to deplete the candidate pattern until an acceptable power distribution is found. Consequently, labor and computer time dedicated to low leakage core reload design can be costly using conventional programs, and the additional resources devoted to determine an adequate cycle could be spent upon finding one more attractive economically. Fortunately, methods have been developed which solve the core depletion calculations in reverse, that is, the core reactivity distribution may be determined given a desired power distribution (Chao,1986;Downar,1986). Such a solution allows the fuel management expert to tailor bundles or burnable

poisons to the predicted core with the capability of solving for patterns both in the forward and reverse depletion directions. Search time is therefore reduced since a degree of backtracking within the cycle calculation is possible. Problem solving procedures for low leakage designs may be further simplified by considering the burnable absorber concentration and assembly positioning search tasks separable (Downar,1986). Although the fuel depletion rate differs during the cycle when there are no burnable absorbers loaded, the accumulated exposure is essentially identical. If an acceptable pattern can be obtained at EOC, then generally a poison strategy exists to control the loading within acceptable power peaking limits throughout the cycle. Conventional loading strategies may then be used at first to propose a loading free of external poisons. Then, with burnable absorbers included in the preliminary loading, burnable and soluble poisons may be adjusted in both the forward and backward depletion directions from the exposure the cycle local power peak occurs. Some additional adjustment of the bundle positioning may be required for an optimal loading once the local power variation throughout the cycle has been minimized by poison composition.

An AI application for low leakage fuel management

remains a current research topic (Rothleder,1988).

VIII.B Performance index

A performance index may be calculated to assist the fuel management expert evaluate costs to realize the design. The index can be a function of fuel costs, the number of new fuel bundles required, the number of bundle moves during refueling, etc. The user would specify the desired weighing factor for each variable.

VIII.C End-of-cycle scoping

This option would analyze the suitability of a loading pattern to meet the energy requirements of the cycle. The analysis, independent of the automated loading search, would deplete a loading to its end-of-cycle for the pattern investigated. Either the core k-effective value for a specified cycle energy production or the maximum possible cycle length for a given loading would be returned.

VIII.D Fuel economics

A fuel economics option could evaluate the fuel cycle costs for a given loading based on fabrication, storage, and materials schedules for fresh fuel bundles. Based on the utility's cash flow rate, then, a net

$/MBtu value could be calculated.

VIII.G Report generation

To be able to distribute, record, and verify the results of a cycle study, a report generation facility capable of producing a hardcopy document of the results including tables, diagrams, and graphs is necessary. This option could report the final loading bundle configuration, neutronics, power, and cycle life, the results of any scoping studies, an explanation of the strategy and criteria applied, the contents of the fuel pools, or produce a work order to perform the shuffle. The report generator would require the following interface capabilities to implement external word processing, spreadsheet, or graphics software the users are familiar with

1. Read, write, and modify data in files of spreadsheet format and create such files for output.
2. Search, replace, and add text to word processor files.
3. Index and access an encyclopedia of files which act as templates to be modified and merged as components to a report document.

VIII.F Summary

The creation, testing, and validation of a problem solving system such as the core reload problem is an evolutionary process and the best advice given a rational design of a knowledge based system is to prepare it for change and evolution. It is likely that between implementation and validation that a more appropriate problem solving environment may be found. Problem solving environments which fully integrate various computational algorithms and programming languages with a hybrid environment are emerging as research tools (Kant,1988). In any case a concise written representation of the problem and an expert's strategy is of value to possess in preparation for transition.

IX. CONCLUSIONS

The choice of using an expert system for in-core fuel management has potentially large rewards or large penalties. If the advice of the expert system proves of adequate help, the rewards may be longer fuel cycles and less downtime between cycles, resulting in increased plant availability. The expert system Shuffle was developed to assist in loading pattern determination for PWRs and combines the following features:

1. An object-oriented representation which simplifies expression of components of the core reload problem and the constraints and rules which constitute its intelligence.
2. A highly interactive environment for display and modification of the solution.
3. External execution of a two group power calculation written in FORTRAN to evaluate loading configurations as they are applied.
4. A concise knowledge base of rules and constraints prepared to solve the core reload problem which documents the problem solution strategies to minimize local power peaking.
5. A weak search method which directs the search toward a near optimum loading with or without an intelligent knowledge base.

The knowledge of Shuffle was evaluated based on its rate of convergence to an improved pattern and its ability to correct poor fuel distributions. Experiments with a number of test patterns showed that additional constraints may improve solution convergence but

restrict solution exploration. Based on its performance some improvements in prediction of fuel placement was suggested such as an initial search with one eighth core symmetry or a choice for an alternative weak search method.

The final evaluation of the Shuffle expert system does not end with this report but should continue as more core loading applications are attempted. It is hoped that Shuffle will reach further development and users.

REFERENCES

Ahn, D. H. and S. H. Levine. "Direct placement of fuel assemblies using the gradient projection method," Transactions of the American Nuclear Society, 46 (June 1984), 123.

Butler, C. W., E. D. Hodil, and G. L. Richardson. "Building knowledge based systems with procedural languages," IEEE Expert, Summer 1988, 47.

Chao, Yung-An, Chaun-Wen Hu, and Chang-An Suo. "A theory of fuel management via backward diffusion calculation," Nuclear Science and Engineering, 93 (1986), 78.

Chen, Yuu-Fuu. J. O. Mingle, and N. D. Eckoff. "Optimal power profile fuel management," Annuals of Nuclear Energy, 4 (1977), 407.

Chitkara, Krishan and Joel Weisman. "An equilibrium approach to optimal in-core fuel management for pressurized water reactors," Nuclear Technology, 24 (October 1974) 33.

Clancy, William J. "Heuristic classification," Artificial Intelligence, 27 (1985), 289.

Colletti, J. P., S. H. Levine, and J. B. Lewis. "A gradient ascent method for determining optimal depletion strategies in PWRs," Transactions of the American Nuclear Society, 39 (November 1981), 473.

Coleman, T. A., J. B. Andrews II, J. R. Worsham III, and V. O. Uotinen. "Improved fuel management with extended burnup fuels," Transactions of the American Nuclear Society, 33 (November 1979), 393.

Comes, Scott A. and P. J. Turinsky. "Out-of-core fuel cycle cost minimization using Monte Carlo integer programming," Topical Meeting on Advances in Fuel Management, Pinehurst, N. C., April 30 1986, ANS, 533.

Dauboin, Pascal. "A knowledge based system for PWR loading pattern determination," Conference on Expert Systems Applications for the Electric Power Industry, June 5-8, 1989 Orlando, Florida.

Downar, Thomas J. and Young Jin Kim. "A reverse depletion method for pressurized water reactor design," Nuclear Technology, 73 (April 1986), 42.

Downar, Thomas J. "LWR core burnup analysis with the

microcomputer," Transactions of the American Nuclear Society, 52 (June 1986), 97.

Ernst, George W. and Ranan B. Banerji. "On the relationship between strong and weak problem solvers," AI Magazine, Summer 1983, 25.

Faught, W. S. Functional Specifications for AI Software Tools for Electric Power Applications, Electric Power Research Institute (EPRI NP-4141), August 1985.

Federowicz, A. J. and R. L. Stover. "Optimization of PWR loading patterns by reference design perturbations," Transactions of the American Nuclear Society, 17 (1973), 308.

Fikes, Richard and Tom Kehler. "The role of frame-based representation in reasoning," Communications of the ACM, 28 (September 1985), 904.

Fikes, R. E. and N. J. Nilsson. "STRIPS: A new approach to the application of theorem proving to problem solving," Artificial Intelligence, 2 (1971).

Galperin A., S. Kimihi and M. Segev. "A knowledge based system for optimization of fuel reload configurations," Nuclear Science and Engineering, 102 (1989), 43.

Haq, S. and P. J. Turinsky. "Out-of-core fuel management optimization utilizing the integer programming technique," Transactions of the American Nuclear Society, 50 (November 1985), 95.

Hayes-Roth, Frederick. "Rule based systems," Communications of the ACM, 28 (September 1985), 921.

Ho, Alfred L. B. and Alexander Sesonske. "Extended burnup fuel cycle optimization for pressurized water reactors," Nuclear Technology, 58 (September 1982), 422.

Ho, Li-Wei, and Alfred F. Rohach. "Perturbation theory in nuclear fuel management optimization," Nuclear Science and Engineering, 82 (1982), 151.

Hobson, Gregory H. and Paul J. Turinsky. "Automatic determination of PWR core loading patterns that maximize core reactivity," Transactions of the American Nuclear Society, 45 (Winter 1983), 96.

Hoshino, Tsutomu. "Optimum fuel loading and operation planning for light water reactor power stations. Part 1: Pressurized water reactor study," Nuclear Technology, 39

(June 1978), 46.

Hoshino, Tsutomu. "In-core fuel management optimization by heuristic learning technique," Nuclear Science and Engineering, 49 (1972), 59.

Hove, C. M., G. C. Robichaud, and J. S. Tulenko. "PWR once-through fuel cycle improvements," Transactions of the American Nuclear Society, 30 (November 1978), 275.

Howland, H. R., C. N. Dorny, and R. L. Stover. "Power flattening and loading pattern searches for pressurized water reactors," Transactions of the American Nuclear Society, 16 (June 1973), 170.

Huang, H. Y., and S. H. Levine. "An automated optimal multiple cycle PWR fuel management code," Transactions of the American Nuclear Society, 30 (November 1978), 338.

Jonsson, C-A. and S. Helmerson. "BWR refueling optimization with ANALOAD," Topical Meeting on Advances in Fuel Management, Pinehurst, N. C., April 30 1986, ANS, 518.

Kant, Elaine. "Interactive problem solving using task configuration and control," IEEE Expert, Winter 1988, 36.

Kawai, Toshio and Takashi Kiguchi. "Optimum refueling order for minimum local power peaking factor," Nuclear Science and Engineering, 43 (March 1971), 342.

Kobayashi, Yasuhiro, Shunsuke Kondo, and Yasumasa Togo. "Optimization of refueling scheme of fast reactor core by means of man machine interactive method," Journal of Nuclear Science and Technology, 14 (March 1977), 177.

Kobayashi, Yasuhiro, Shunsuke Kondo, and Yasumasa Togo. "Optimization of refueling scheme of sodium-cooled fast reactor core using heuristic enumeration method," Journal of Nuclear Science and Technology, 13 (September 1976), 471.

Kunz, John C., Thomas P. Kehler and Michael D. Williams. "Applications development using a hybrid AI development system," AI Magazine, Fall 1984, 41.

Lin, Bo-In., Burt Zolotar, and Joel Weisman. "An automatic procedure for selection of optimal refueling policies for light water reactors," Nuclear Technology, 44 (July 1979), 258.

Long, Alexander B., and Joseph M. Holzer. "The EASE+CORE system for fuel cycle management," Transactions of the American Nuclear Society, 52 (June 1986), 101.

Menzies, Tim. "Domain-specific knowledge representation," AI Expert, June 1989, 36.

Mingle, John O. "Nuclear fuel management via fuel quality factor averaging," Annuals of Nuclear Energy, 5 (1978), 645.

Mingle, John O. "In-core fuel management via fuel perturbation theory," Nuclear Technology, 27 (October 1975), 248.

Morita, T., Y. A. Chao, A. J. Federowicz, and P. J. Duffy. "LPOP: Loading Pattern Optimization Program," Transactions of the American Nuclear Society, 52 (June 1986), 41.

Motoda, Hiroshi., John Herczeg, and Alexander Sesonske. "Optimization of refueling schedule for light water reactors," Nuclear Technology, 25 (March 1975), 477.

Motoda, Hiroshi. and Osamu Yokomizo. "Optimization of fuel assembly allocation for boiling water reactors," Journal of Nuclear Science and Technology, 13 (May 1976) 230.

Naft, Barry N. and Alexander Sesonke, "Pressurized water reactor optimal fuel management," Nuclear Technology, 14 (May 1972), 123.

O'Hare, G. M. and D. A. Bell. "The coexistence approach to knowledge representation," Expert Systems, 2 (October 1985), 230.

Pearl, Judea. Heuristics: Intelligent Search Strategies for Computer Problem Solving, Addison-Westley Publishing Co., 1984, 35.

Prerau, David S. "Selection of an appropriate domain for an expert system," AI Magazine, Summer 1985, 26.

Ramamoorthy, C. V. and Philip C. Sheu. "Object-oriented systems," IEEE Expert, Fall 1988, 9.

Robichaud, G. G. and J. S. Tulenko. "Core design: Fuel management as a balance between in-core efficiency and economics," Power, May 1978, 52.

Robinson, Alan H., J. O. Heaberlin, and G. L. Wang. "An automated search procedure for fuel shuffling in PWRs including rotation effects," ANS Topical Meeting on Artificial Intelligence and Other Innovative Computer Applications in the Nuclear Industry: Present and Future, Snowbird, UT, August 1987.

Rothleder, B. M., and W. S. Faught. "A prototype fuel-shuffling system using a knowledge based tool kit," ANS Topical Meeting on Computer Applications for Power Plant Operation and Control, Pasco, Washington, September 8-12, 1985, American Nuclear Society (1986), 738.

Rothleder, B. M., G. R. Poetschat, and W. S. Faught. "The potential for expert system support in solving the pressurized water reactor fuel shuffling problem," Nuclear Science and Engineering, (100) 1988, 440.

Sekimizu, Koichi. "A hierarchy level scheme for quasi-optimum fuel assembly loading in boiling water reactors," Nuclear Technology, 37 (March 1978), 296.

Simon, Herbert A. "Search and reasoning in problem solving," Artificial Intelligence, 21 (1983), 7.

Smith, D. E., L. F. Kocher, and S. E. Seeman. "CLEO: a knowledge-based refueling assistant at FFTF," ANS Transactions, 50 (1985), 292.

Stone, Jeffery. "The AAAI-86 conference exhibits: new directions for commercial AI," AI Magazine, Spring 1987, 49.

Story, G. T. and R. L. Grow. "A microcomputer workstation for design of PWR fuel loading patterns," Transactions of the American Nuclear Society, 50 (1985), 97.

Stout, Richard B. Optimization of In-Core Nuclear Fuel Management in a Pressurized Water Reactor, Phd Thesis, Oregon State University, 1972.

Stout, Richard B. and A. H. Robinson. "Determination of optimum fuel loadings in pressurized water reactors using dynamic programming." Nuclear Technology, 20 (November 1973), 86.

Stillman, J. A., Y. A. Chao, and T. J. Downar. "A two dimensional fuel-loading optimization method for the PWR burnup cycle," Transactions of the American Nuclear Society, 59 (June 1989), 351.

Takenouchi, H. and Y. Iwashita. "An integrated knowledge representation scheme for expert systems," Expert Systems, 4 (February 1987), 38.

Terney, William B. and E. A. Williamson, Jr. "The design of reload cores using optimal control theory," Transactions of the American Nuclear Society, 27 (November 1977), 361.

Uhrig, Robert E. and Lawence F. Miller. "Use of neural networks for in-core fuel managment," Transactions of the American Nuclear Society, 59 (June 1989), 59.

Wang, G. L. and A. H. Robinson. An Automatic Procedure for Optimizing the Refueling of Pressured Water Reactors which includes Assembly Rotation Effects, Oregon State University (OSU-NE-8701), February 1987.

Williamson. E. A., Jr., William B. Terney, and Dennis J. Huber. "Interactive fuel management using a CRT," Transactions of the American Nuclear Society, 30 (November 1978), 341.

APPENDICIES

APPENDIX A

NODAL POWER CALCULATION

A.1 Abstract

The core power calculation code, PWRCALC, interfaces with the core reload system to answer the relative power distribution for the desired loading configuration, burnup, and boron shim. Written in FORTRAN77, the code is executed external to the expert system environment and has no special requirements upon the FORTRAN compiler used to create its object file. Data between the expert system and the core power calculation is communicated in the form of ASCII text files in free format and a library file is required to reference cross section correlations by burnup for each generic bundle type. The calculation is based on the two group two dimensional diffusion equations solved with an ADI inner iteration and SOR outer iteration. Variable material properties are modeled. This appendix serves as reference for the code's interface requirements, representation, models, and verification.

A.2 Program Capabilities

1. Material properties may vary by cell.
2. Borresen flux averaging is employed.

3. A single bundle is represented by four nodes.

4. The power calculation accounts for soluble boron concentration and desired burnup.

5. Cross sections are represented as a correlation of fuel exposure.

6. Cross section correlations are indexed by bundle class and are read from a library file which the user is free to modify or supplement.

7. The loading pattern is constructed from input as the bundle type, coordinate position, and quadrant exposure is read. This enables the configuration to be changed easily.

8. The code checks for bundles interior to the loading and reports an error if one or more is missing.

9. The two group two dimensional diffusion equations are solved by ADI inner iteration and SOR outer iteration.

10. An optional search for the boron concentration for

criticality may be performed.

A.3 Requirements

Two input files are needed by PWRCALC in the directory where it executes--BUNDLES.DAT and BUNDLES.LIB. In addition boron and water cross sections and some parameters specific to the solution are coded into the program. This section discusses the preparation of these input files, identifies the coded parameters, and discusses the format of the single output file generated, BUNDLES.PWR.

A.3.1 BUNDLES.DAT Input format

If the entry for soluble boron is negative a search is performed and the concentration of boron required to keep the loading critical at the specified exposure is answered in BUNDLES.PWR. If the loading is subcritical without boron a concentration of zero is answered. Note also the entry for bundle orientation, IROT, is not used in the problem solution but is only printed in the output file BUNDLES.PWR. It is included to complete the representation of the loading within the file.

File: BUNDLES.DAT

Entries		Description
Rec 1:	nbun	Total number of bundles in the loading.
	ppmb	Loading soluble boron

		concentration (ppm). If negative a criticality concentration search is performed.
	expi	Initial exposure of loading (GWD/MT).
	expf	Final exposure of loading (GWD/MT). If expf <= expi no burn calculation is performed.
	expinc	Burnup increment (GWD/MT). If expinc <= 0 no burn calculation is performed. The final burn step is either expinc or (expf - expi) whichever is least.

Read i = 1 to nbun records of data sets for each bundle.

{Rec i:	K	Column of bundle position.
	L	Row of bundle position.
	bid(L,K)	Bundle identification (enclosed in quotes e.g. 'D23').
	idx(L,K)	Bundle library index integer.
	irot(L,K)	Bundle orientation integer.

Read 4 quadrant exposures for the bundle in the sequence shown in Fig. A.1

	{ex(i,j)	Bundle quadrant exposure}}

A.3.2 BUNDLES.LIB

A.3.2.1 Input format

A data set of eight records appears in the library for each generic bundle type with a bundle library index appearing in record one. All data but the library index

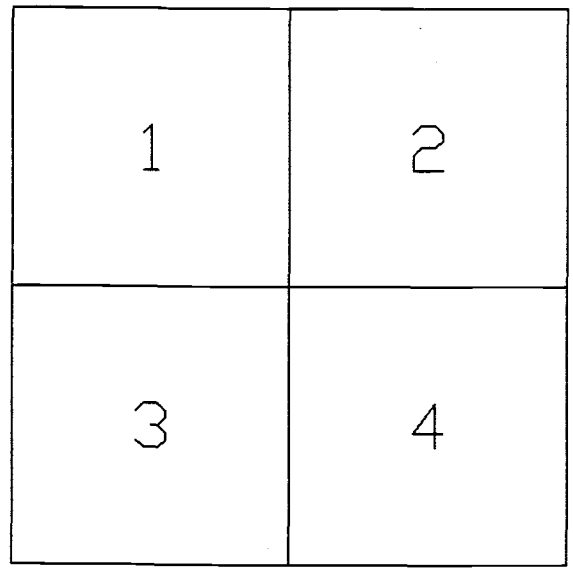

Figure A.1. Bundle quadrant sequence in PWRCALC file interface.

is not read free format. Prior access to the library, all unique cross section indices for bundles in the map are categorized. When the library is accessed, the library index is read from the first record of each set and if it matches one of the indices required the correlation constants are read from the library otherwise the file is positioned forward to the next set. All sets in the library are read until no further library indices required by the loading configuration remain. Cross section constants for each linear correlation of exposure are sequenced by increasing order in the correlation, e.g. for correlation

$$\text{siga(bundle)} = A0 + A1*\exp(\text{bundle}) + A2*\exp(\text{bundle})^2 + A3*\exp(\text{bundle})^3 + A4*\exp(\text{bundle})^4$$

constants A0, A1, A2, A3, and A4 appear in sequence left to right on each record.

File: BUNDLES.LIB

Entries		Description
Read for all entries of idx in BUNDLES.LIB		
{Rec i:	idx	Bundle library index integer (I2 format)
Read j = 1 to 5 constants for fast absorption		
{Rec i+1:	rcxa1(j)	Bundle linear correlation for fast absorption}
Read j = 1 to 5 constants for thermal absorption		
{Rec i+1:	rcxa2(j)	Bundle linear correlation for thermal absorption}

```
Read j = 1 to 5 constants for fast nu-fission
{Rec i+1: rcxf1(j)        Bundle linear correlation for
                          fast nu-fission}

Read j = 1 to 5 constants for thermal nu-fission
{Rec i+1: rcxf2(j)        Bundle linear correlation for
                          thermal nu-fission}

Read j = 1 to 5 constants for the slowing cross section
{Rec i+1: rcxsc(j)        Bundle linear correlation for
                          slowing cross section}

Read j = 1 to 5 constants for fast diffusion coefficient
{Rec i+1: rcxd1(j)        Bundle linear correlation for
                          fast diffusion}

Read j = 1 to 5 constants for thermal diffusion
coefficient
{Rec i+1: rcxd2(j)        Bundle linear correlation for
                          thermal diffusion}}
```

A.3.2.2 Correlation preparation

Entries of the library consist of cross sections in fourth order linear correlations by exposure indexed by bundle class. LEOPARD burn runs from zero to the maximum rated exposure of each assembly generate the cross sections (Barry,1964). All cross sections but those prior equilibrium xenon buildup are then used to prepare the correlations. Typically this means only cross sections at exposures above 0.5 GWD/MT are correlated.

Eight fuel types, C, D, G, H, J, K, L, and M, are implemented in the Trojan Cycle 10 loading which was used as reference for the simulation. To reduce the number of LEOPARD runs required to generate cross sections, bundles with similar enrichments were grouped

Table A.1: Bundle assignment for library creation.

PGE (Type	Enrich)	OSU (Type	Enrich)	Index
C	3.088	C	3.09	1
D	3.098	D	3.10	2
G	3.198	G	3.20	3
H	3.296	HJ	3.30	4
J	3.296			
K	3.451	KL	3.43	6
L	3.417			
M	3.394	M	3.39	5

Table A.2: Bundle concentrations in extra region.

Bundles C, D, HJ, KL, and M

Element	Concentration
Ziralloy 2 (3)	.02634
Iron (6)	.000262
Niobium (7)	.0007814
Chromium (11)	.0002266
Water (100)	.9724
Non-lattice fraction	.0865

Bundle G (with 5 steel rods)

Element	Concentration
Ziralloy 2 (3)	.02248
Iron (6)	.000224
Niobium (7)	.000667
Chromium (11)	.000194
SS-304 (304)	.066475
Water (100)	.90997
Non-lattice fraction	.1014

so that only six fuel types needed to be represented. Table A.1 shows the groupings implemented. To model each bundle accurately, a number of structural Table elements were included in the extra region of LEOPARD's input data and are shown in Table A.2 by bundle class. All bundles have similar structural elements except for type G which has five steel rods in place of fuel pins. Standard operating conditions of Trojan such as coolant temperature and pressure were also required in the LEOPARD input. A sample LEOPARD input file is shown in Figure A.2. LEOPARD was run for each fuel type over 14 intervals to an exposure of 40 GWD/MT with intervals reduced near zero exposure for sufficient detail. The two group macroscopic cross sections generated above 0.1 GWD/MT were then correlated for each fuel. As a test, selected correlations for one fuel type are plotted against the original cross sections from LEOPARD to show agreement for slowing, thermal absorption, and thermal fission in Figures A.3 to A.5 for one fuel type. It can be seen that the data agrees closely with the correlations. The resulting library file for PWRCALC is shown in Figure A.6.

A.3.3 BUNDLES.PWR Output format

File: BUNDLES.PWR

Entries	Description

```
TYPE C FUEL (3.09%) WITH BURN TO 40000 MWD PER MT
 1  0  0  0   0  0  1  0   0  0  0  0  1  1  0  0   0  0  1
        99  1.0          0.0          0.0          0.0
         3  0.0          1.0          0.0          0.02634
         6  0.0          0.0          0.0          0.000262
         7  0.0          0.0          0.0          0.0007814
        11  0.0          0.0          0.0          0.0002266
       100  0.0          0.0          1.0          0.9724
       777  0.0          0.0          0.0          0.0
        18 -0.0309
       777  0.0
 970.00000  1070.00000   623.000000   585.000000   0.0000010    1.10000000
0.40950000  0.47498000   1.25984000   1.00000000   375.60000    0.08650000
2250.00000  0.0          10.2000000   0.0          0.0          0.4
1.00000000  104.000000   0.0          0.0          1.0000000
         1  -20.000000
         2  -80.000000
         3  -400.00000
         4  -500.00000
         5  -1000.0000
         6  -2000.0000
         7  -3000.0000
         8  -5000.0000
         9  -5000.0000
        10  -5000.0000
        11  -5000.0000
        12  -5000.0000
        13  -5000.0000
        14  -5000.0000
        15  -5000.0000
       777  0.00000000
```

Figure A.2. LEOPARD input file to generate bundle cross sections.

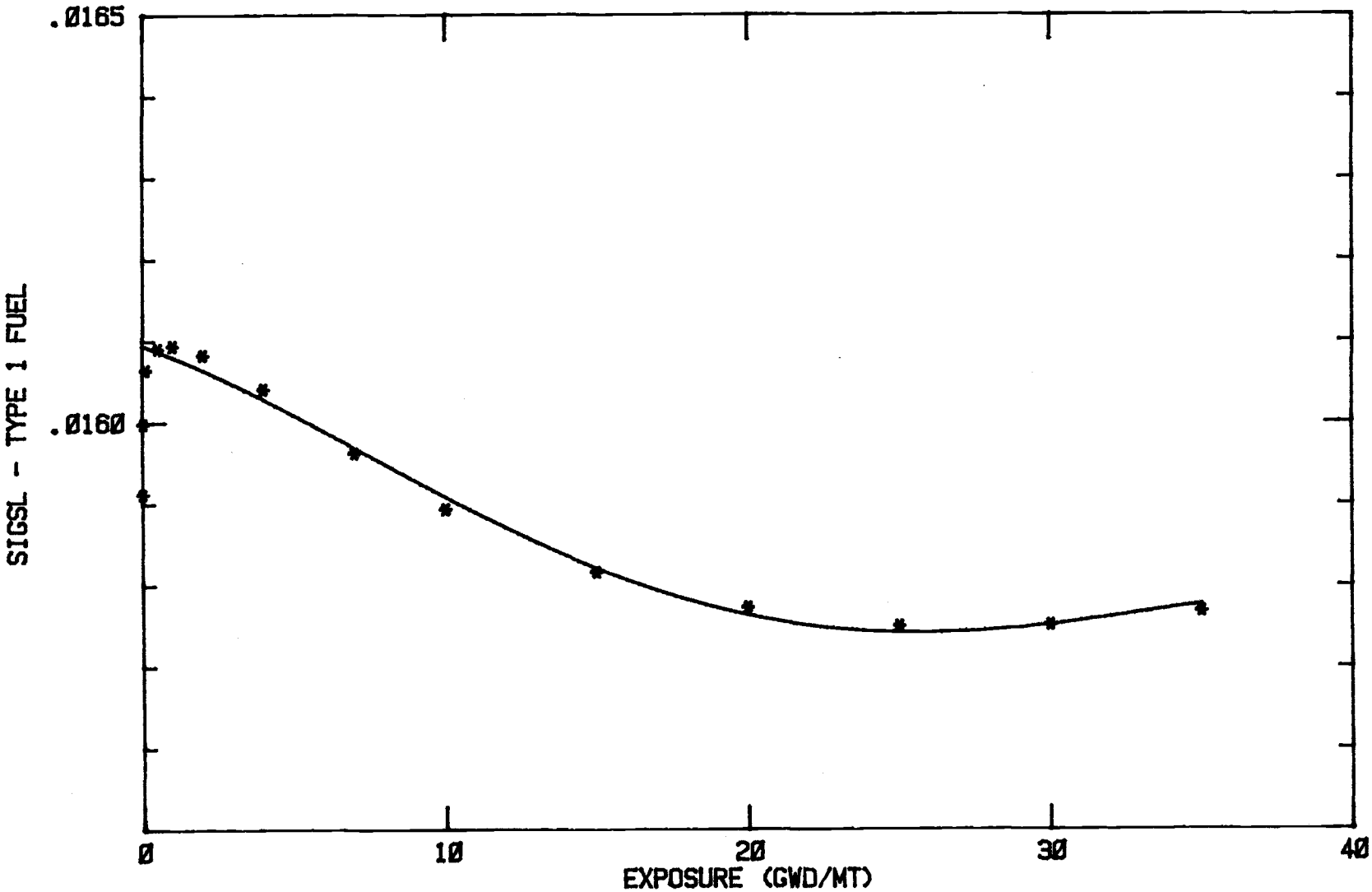

Figure A.3. Performance of slowing cross section correlation for bundle index 1.

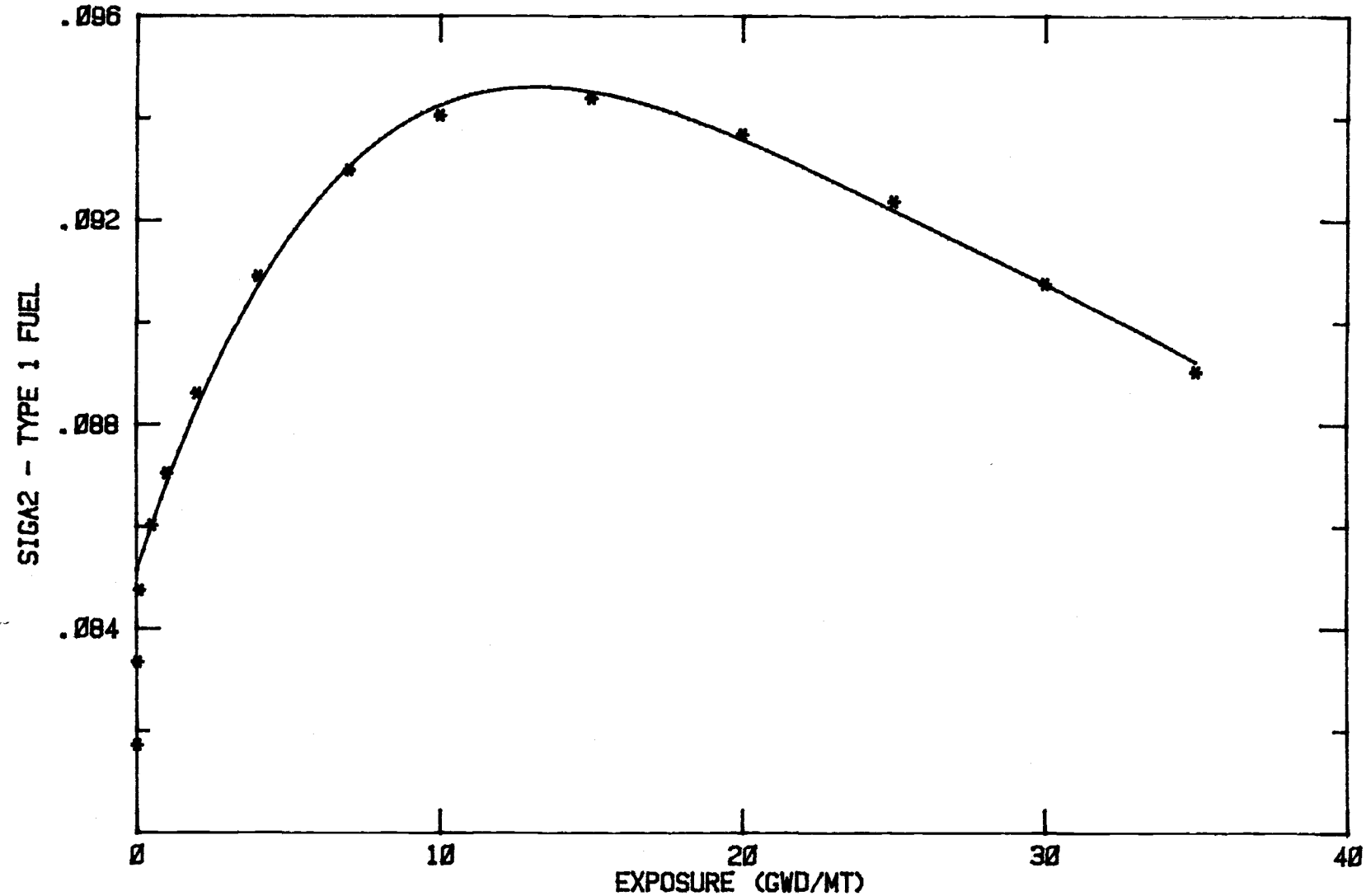

Figure A.4. Performance of thermal absorption cross section correlation for bundle index 1.

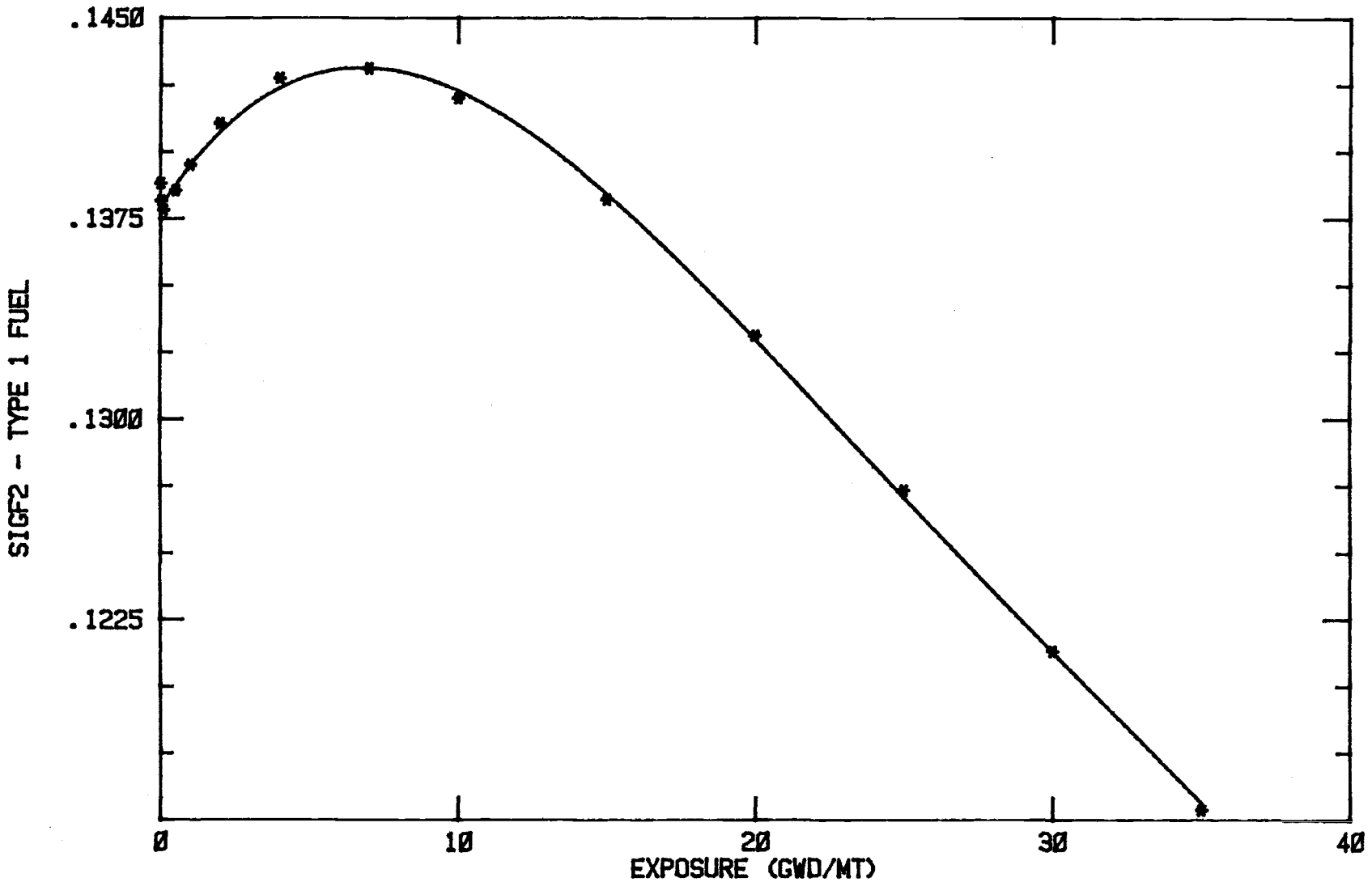

Figure A.5. Performance of thermal fission cross section correlation for bundle index 1.

```
1   Type C (3.09%)
  .915130E-02  .928211E-04  .347050E-06 -.635813E-07  .885753E-09
  .851520E-01  .180757E-02 -.115334E-03  .281504E-05 -.257271E-07
  .650474E-02 -.570425E-04 -.431061E-06  .254554E-07 -.266051E-09
  .137955E+00  .170122E-02 -.163775E-03  .404189E-05 -.363388E-07
  .160938E-01 -.130140E-04 -.107458E-05  .596490E-07 -.734873E-09
  .142722E+01  .884046E-03  .169165E-03 -.731642E-05 -.746748E-07
  .385764E+00 -.131386E-02  .661298E-04 -.158716E-05  .146662E-07
2   Type D (3.10%)
  .915757E-02  .926315E-04  .342785E-06 -.630965E-07  .877042E-09
  .853323E-01  .180518E-02 -.115163E-03  .280910E-05 -.256684E-07
  .651728E-02 -.571082E-04 -.434454E-06  .255497E-07 -.267489E-09
  .138324E+00  .169644E-02 -.163415E-03  .402870E-05 -.362028E-07
  .160860E-01 -.126846E-04 -.110867E-05  .611968E-07 -.757819E-09
  .142718E+01  .893826E-03  .167010E-03 -.721543E-05  .735125E-07
  .385756E+00 -.131280E-02  .660160E-04 -.158354E-05  .146285E-07
3   Type G (3.20%) with 5 steel rods
  .912169E-02  .881336E-04  .480552E-06 -.657274E-07  .898779E-09
  .867412E-01  .173169E-02 -.111009E-03  .269191E-05 -.245479E-07
  .654725E-02 -.578908E-04 -.420994E-06  .245491E-07 -.255516E-09
  .139432E+00  .158705E-02 -.156198E-03  .381216E-05 -.340766E-07
  .161199E-01 -.112750E-04 -.111794E-05  .603973E-07 -.736892E-09
  .142311E+01  .838423E-03  .164709E-03 -.706246E-05  .712130E-07
  .384131E+00 -.126359E-02  .629330E-04 -.149936E-05  .138022E-07
4   Type H/J (3.30%)
  .928361E-02  .879324E-04  .447218E-06 -.637339E-07  .869025E-09
  .889038E-01  .176004E-02 -.111900E-03  .269746E-05 -.245836E-07
  .676701E-02 -.585298E-04 -.464332E-06  .255271E-07 -.266189E-09
  .145636E+00  .160556E-02 -.156452E-03  .377780E-05 -.336526E-07
  .159352E-01 -.920409E-05 -.129813E-05  .666495E-07 -.812990E-09
  .142623E+01  .129277E-02  .899733E-04 -.348856E-05  .240552E-07
  .385582E+00 -.129202E-02  .638335E-04 -.151492E-05  .139246E-07
5   Type M (3.39%)
  .934008E-02  .859125E-04  .500780E-06 -.645482E-07  .873656E-09
  .904895E-01  .174149E-02 -.110543E-03  .265200E-05 -.241506E-07
  .687858E-02 -.591287E-04 -.472590E-06  .253095E-07 -.262953E-09
  .148885E+00  .156770E-02 -.153492E-03  .367301E-05 -.326014E-07
  .158696E-01 -.802168E-05 -.133522E-05  .669778E-07 -.809730E-09
  .142585E+01  .146314E-02  .568405E-04 -.187091E-05  .244957E-08
  .385494E+00 -.128266E-02  .628917E-04 -.148573E-05  .136269E-07
6   Type K/L (3.43%)
  .936511E-02  .850457E-04  .523177E-06 -.648730E-07  .875302E-09
  .911901E-01  .173354E-02 -.109957E-03  .263255E-05 -.239667E-07
  .692800E-02 -.593846E-04 -.475935E-06  .252085E-07 -.261521E-09
  .150320E+00  .155136E-02 -.152201E-03  .362761E-05 -.321480E-07
  .158407E-01 -.752342E-05 -.135056E-05  .671127E-07 -.808471E-09
  .142569E+01  .153712E-02  .424513E-04 -.116840E-05 -.692842E-08
  .385453E+00 -.127849E-02  .624796E-04 -.147303E-05  .134978E-07
```

Figure A.6 PWRCALC library file for bundles of Cycle 10.

Write i = 1 to nBun records of data regarding each bundle at the characteristic exposure of the loading.

{Rec i:	bid(L,K)	Bundle identification.
	K	Column of bundle position.
	L	Row of bundle position.
	idx(L,K)	Bundle library index integer.
	irot(L,K)	Bundle orientation integer.

Write 4 quadrant exposures for the bundle in the sequence shown in Fig. A.1

	{ex(i,j)	Bundle quadrant exposure}

Write 4 quadrant reactivities for the bundle in the sequence shown in Fig. A.1

	{re(i,j)	Bundle quadrant reactivity}

Write 4 quadrant powers for the bundle in the sequence shown in Fig. A.1

	{rpow(i,j)	Bundle quadrant power}}

Rec: nbun+1	effk	Loading effective multiplication factor.
	ppmb	Loading soluble boron concentration (ppm).
	expi	Exposure of the loading generated.
	Kmax	Column of maximum power bundle position.
	Lmax	Row of maximum power bundle position.

A.3.4 Boron Concentration

In addition to the correlations, the cross sections calculated for each node must account for the boron concentration in the reactor coolant. Boron-10 is a

strong absorber of thermal neutrons and its presence in the coolant helps to maintain reactivity of the system without disturbing the power distribution in the core. Although the shim is added in the form of boric acid (H_3BO_3) to the coolant, only the thermal absorption cross section of boron-10 is accounted since it has the most pronounced effect on the overall node cross section. The concentration of boric acid is usually specified is parts per million (ppm) of water which implies a measure of one gram of boron per million grams water. In terms of the number of atoms of boron-10 to water this is

$$\frac{N_B}{N_W} = .198 \frac{M_W}{M_B} C \times 10^{-6}$$

where

N_B = number density of boron-10
N_W = number density of water
M_W = atomic mass of water
M_B = atomic mass of boron-10
C = concentration of H_3BO_3 in water (ppm)

and the expression has been adjusted to represent the amount of boron-10 which naturally occurs in boron (19.8%). The macroscopic thermal absorption cross section of boron-10 by

$$\Sigma_{aB} = \sigma_B N_B$$

is then added to the overall thermal absorption cross

section for the node where σ_B is the microscopic absorption cross section of boron-10 (2087.445 b). The number density of water may be obtained by one of two methods: for nodes represented by bundles in the map this is one half the volume weighted number density of hydrogen from LEOPARD; for nodes in the moderator the number density is

$$N_W = \frac{\rho N_a}{M_W}$$

where

ρ = density of water (g cm^{-3})
N_a = Avogadro's number
(.6022 atoms cm^2 $mole^{-1}$ b^{-1}).

A.3.5 Reactivity Calculation

The k-infinity of each bundle is calculated from the two group expression

$$k_\infty = \frac{\nu\Sigma_{f1} + \nu\Sigma_{f2}\frac{\Sigma_{s112}}{\Sigma_{a2}}}{\Sigma_{r1}}$$

given the cross sections from the library file calculated for the bundle exposure and where

Σ_{r1}= Macroscopic removal cross section for group L.

Σ_{a2}= Macroscopic thermal absorption cross section.

Σ_{s112}= Macroscopic slowing cross section from energy group 1 to 2.

$\nu\Sigma_{fL}$=Fission source cross section for group L=1,2.

A.3.6 Coded parameters

Table A.3 summarizes the coded parameters of PWRCALC which may need to be altered if operating parameters of the reactor, the bundle dimensions, the shim and coolant materials, the Borresen averaging factors, or the source convergence require change.

A.4 Representation

The lower right quarter of a full sized core loading is modeled by PWRCALC such that bundle placement is identified by row and column from the core center. Each bundle quadrant in the map is associated with a node in the power calculation. Two coordinate systems are used to identify position of a bundle quadrant; a relative indexing scheme which identifies the quadrant position to the bundle and an absolute indexing scheme to identify the quadrant position by map row and column. Both coordinate systems are needed to satisfy interface requirements between the object oriented expert system which uses relative indexing and the nodal power calculation which needs absolute indexing. Figure A.7 attempts to illustrate the bundle, absolute quadrant, and relative quadrant indexing systems. If x@y symbolizes a point with x as the row coordinate position

Table A.3: Coded parameters

Parameter	Value	Description
maxo	40	Maximum number of outer iterations.
testot	1e-6	Convergence error on outer iteration.
alphao	1.7	Acceleration factor.
size	10.5cm	Bundle width / 2.
aa(1)	0	Fast Borresen factor.
aa(2)	0	Thermal Borresen factor.
xnhyd	.02938342	Volume weighted number density of hydrogen from LEOPARD.
cxbmic	2087.445b	B-10 average thermal microscopic cross section at moderator temperature.
h2oden	.711611gcm^{-3}	Density of water at operating temperature and pressure.
sigslw	.342525E-1cm^{-1}	Slowing down cross section of water from LEOPARD
dwater(1)	1.911565cm^{-1}	Macroscopic fast diffusion coefficient of water.
dwater(2)	.2866381cm^{-1}	Macroscopic thermal diffusion coefficient of water.
sigaw(1)	.568668E-3cm^{-1}	Macroscopic fast absorption cross section of water.
sigaw(2)	.969337E-2cm^{-1}	Macroscopic thermal absorption cross section of water.

and y as the column coordinate position the following relations may be used to convert between coordinate systems.

$$\text{absolute quadrant} = 2(\text{bundle position} - 1@1) + \text{relative quadrant}$$

$$\text{relative quadrant} = \text{mod}(\text{absolute quadrant}, 2)$$

$$\text{bundle position} = \text{absolute quadrant} / 2$$

where each parameter is a point, the third equation uses integer division, and the modulus function behaves like the FORTRAN function of the same name.

Figure A.7a shows the locations of the reflected and extrapolated boundaries in the map. Also, beyond the face of bundles on the map periphery an additional node is added in the moderator to improve accuracy of the nodal calculation due to the sharp flux gradient there.

A.5 Nodal Calculation

A.5.1 Requirement for optimization

Originally, the core power calculation implemented by Richard Stout represented one node per bundle and used successive displacement on the flux or inner iteration and successive overrelaxation (SOR) on the source or outer iteration. Later the code was modified

to represent four nodes per bundle and ran on a microcomputer as opposed to a high speed mainframe. The results of the power calculation were now highly dependent on flux convergence and required a test on at least five different locations in the map to verify convergence. Unfortunately, if the ratio of inner iterations per outer iteration or the convergence criteria was changed the location of the peak power in the map would shift. Because the number of trial loadings the expert system can generate is directly proportional to the speed of the core power calculation and its accuracy is dependent on the calculation as well, an attempt was made for its improvement.

A.5.2 Summary of optimization experience

E. L. Wachspress studied optimal methods for solving elliptic equations such as the diffusion equation and arrived at some generalizations regarding SOR, Chebyshev extrapolation, and ADI iteration methods (Wachspress,1966). He concluded that a significant time savings could be realized by extrapolation of the outer iteration and by a balance between the number of inner and outer iterations. If the problem size to be solved is large, the computation time may vary primarily as the total number of inner iterations. In such a case it is recommended to perform relatively few inner iterations

per outer. Alternatively, if computation time depends more strongly upon the number of outer iterations it is then best to perform enough inner iterations per outer iteration to yield a significant error reduction. Wachspress suggests Chebyshev extrapolation of the outer iteration may be used effectively when the error measure of the inner iteration is reduced by a factor of 10%.

For nodal problems of larger dimension, then, an optimal calculation depends on reducing the calculation time of the inner iteration. Wachspress examined convergence rates realized for ADI iteration or SOR iteration when solving Poisson's equation over a rectangular region--the same formulation the diffusion equation assumes for the core power calculation. He demonstrated mathematically that the ADI method is almost certainly more efficient than SOR and the advantage of ADI becomes more pronounced as the problem size increases. The complexity of coding the ADI iteration compared to SOR, however, is such that SOR is better suited for solving small problems.

In conventional solutions, when the diffusion equation is solved with inner and outer iterations the majority of computation time is required by the inner iteration which essentially calculates the inverse of the diffusion operator. The same diffusion operator is repetitively inverted numerically by the outer

iteration. If instead the inverted diffusion operator could be saved in storage the inner iterations could be eliminated. A technique based on this concept known as the principle of diffusive homogeneity eliminates the inner iteration by rendering the diffusion equation so it contains a unique generic diffusion operator which only needs to be inverted once for each application in the diffusion calculations. Recently a multidimensional nodal method employing the principle of diffusive homogeneity has been reported whose speed is two orders of magnitude faster than conventional nodal codes (Chao,1987).

Experimentation with the ratio of inner iterations to outer iterations with the core power calculation revealed its speed was strongly limited by the inner iteration since reducing the number of inner iterations accelerated convergence. In an attempt to improve the speed and accuracy of the power calculation further modifications were made to replace the inner iteration with the ADI method.

A.5.3 ADI Treatment of Inner Iteration

The two group diffusion equations to be solved in two dimensions appear as

$$-\nabla\cdot D_1(x,y)\nabla\phi_1(x,y) + \Sigma_{r1}(x,y)\phi_1(x,y) = \frac{1}{k}(\nu\Sigma_{f1}(x,y)\phi_1(x,y) + \nu\Sigma_{f2}(x,y)\phi_2(x,y))$$

$$-\nabla\cdot D_2(x,y)\nabla\phi_2(x,y) + \Sigma_{a2}(x,y)\phi_2(x,y) = \Sigma_{s112}\phi_1(x,y) \qquad \text{(A.1)}$$

where

$\phi_L(x,y)$ = Neutron flux for group L.
$D_L(x,y)$ = Diffusion coefficient for group L.
$\Sigma_{rL}(x,y)$ = Macroscopic removal cross section for group L.
= Σ_{a2} if L = 2.
$\Sigma_{a2}(x,y)$ = Macroscopic thermal absorption cross section.
$\Sigma_{s112}(x,y)$ = Macroscopic slowing cross section from energy group 1 to 2.
$\nu\Sigma_{fL}(x,y)$ = Fission source cross section for group L.
k = multiplication factor.
L = energy group (fast: 1, thermal: 2).

These set of elliptic equations may be solved by the ADI method by noting the solution may be considered as the limiting solution of the time dependent problem, i.e.

$$\frac{\partial\phi_1(x,y)}{\partial t}\left(\frac{1}{v_1}\right) = \nabla\cdot D_1(x,y)\nabla\phi_1(x,y) - \Sigma_{r1}(x,y)\phi_1(x,y) + \frac{1}{k}(\nu\Sigma_1(x,y)\phi_1(x,y) + \nu\Sigma_2(x,y)\phi_2(x,y))$$

$$\frac{\partial\phi_2(x,y)}{\partial t}\left(\frac{1}{v_2}\right) = \nabla\cdot D_2(x,y)\nabla\phi_2(x,y) - \Sigma_{a2}(x,y)\phi_2(x,y) + \Sigma_{s112}(x,y)\phi_1(x,y) \qquad \text{(A.2)}$$

where v_g is the neutron velocity for group g.
Here the term k has been introduced since we anticipate the equation will reach the form of the eigenvalue problem. Given an initial condition on flux, equation

set A.2 is solved by successive time steps to arrive at the solution for A.1. The ADI method expresses each of these equations as two difference equations which are used over successive half interval time steps. This enables tridiagonal matrix solutions separately implicit by row (i) or column (j). Assume for illustration the problem is restricted to homogeneous properties. The finite difference equations for the ADI method applied to the fast energy group would then appear as

$$\frac{(\phi^* - \phi^n)_1}{\frac{\Delta t}{2}}\left(\frac{1}{v_1}\right) - D_1\left(\frac{\partial^2\phi_1^*}{\partial x^2} + \frac{\partial^2\phi_1^n}{\partial y^2}\right) + \Sigma_{r1}\phi_1^n = \frac{1}{k}(\nu\Sigma_{f1}\phi_1^n + \nu\Sigma_{f2}\phi_2^n)$$

$$\frac{(\phi^{n+1} - \phi^*)_1}{\frac{\Delta t}{2}}\left(\frac{1}{v_1}\right) - D_1\left(\frac{\partial^2\phi_1^*}{\partial x^2} + \frac{\partial^2\phi_1^{n+1}}{\partial y^2}\right) + \Sigma_{r1}\phi_1^{n+1} = \frac{1}{k}(\nu\Sigma_{f1}\phi_1^{n+1} + \nu\Sigma_{f2}\phi_2^{n+1})$$

where time step superscript n indicates flux prior a time step, n+1 indicates flux following a full time step, and * references flux at one half a time step. Since there are two equations in A.1 coupled by fluxes of each energy group precautions are necessary to solve the set of equations for fluxes at the present time step before advancing to the set at the next half time step.

If we assume a time step increment, Δt, which approaches infinity the time superscripts now represent present and future values in flux iteration. An explicit treatment for the difference equations will come in the following section, however the equations to be solved should appear like the form below

Time step 1

$$-D_1\left(\frac{\partial^2 \phi_1^*}{\partial x^2} + \frac{\partial^2 \phi_1^n}{\partial y^2} \right) + \Sigma_{r1}\phi_1^n = \frac{1}{k^n}(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)^n$$

$$-D_2\left(\frac{\partial^2 \phi_2^*}{\partial x^2} + \frac{\partial^2 \phi_2^n}{\partial y^2} \right) + \Sigma_{a2}\phi_2^* = \Sigma_{s112}\phi_1^*$$

Time step 2

$$-D_1\left(\frac{\partial^2 \phi_1^*}{\partial x^2} + \frac{\partial^2 \phi_1^{n+1}}{\partial y^2} \right) + \Sigma_{r1}\phi_1^{n+1} = \frac{1}{k^*}(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)^*$$

$$-D_2\left(\frac{\partial^2 \phi_2^*}{\partial x^2} + \frac{\partial^2 \phi_2^{n+1}}{\partial y^2} \right) + \Sigma_{a2}\phi_2^{n+1} = \Sigma_{s112}\phi_1^{n+1}$$

where each equation solves fluxes in the following sequence

1. Given ϕ_1^n, ϕ_2^n solve ϕ_1^*
2. Given ϕ_2^n, ϕ_1^* solve ϕ_2^*
3. Given ϕ_1^*, ϕ_2^* solve ϕ_1^{n+1}
4. Given ϕ_2^*, ϕ_1^{n+1} solve ϕ_2^{n+1}

That is, given ϕ^n the solution is performed for the first half time step to determine ϕ^*, then the next half time step is solved from ϕ^* for ϕ^{n+1}. Separate multiplication factors for each time interval are used where

$$k^{n+1} = k^n \frac{\sum_L \nu\Sigma_{fL}\phi_L^{n+1}}{\sum_L \nu\Sigma_{fL}\phi_L^{n}}$$

$$k^* = k^{*-1} \frac{\sum_L \nu\Sigma_{fL}\phi_L^{*}}{\sum_L \nu\Sigma_{fL}\phi_L^{*-1}} \qquad (A.3)$$

So the multiplication factor k is calculated over source values between succeeding time steps.

A.5.4 Difference Equations

A detailed treatment of the diffusion equations in difference form will be presented here. Start with the time dependent diffusion equations ignoring delayed neutrons.

$$\frac{\partial\phi_1(x,y)}{\partial t}\left(\frac{1}{v_1}\right) = \nabla\cdot D_1(x,y)\nabla\phi_1(x,y) - \Sigma_{r1}(x,y)\phi_1(x,y) + \frac{1}{k}(\nu\Sigma_{f1}(x,y)\phi_1(x,y) + \nu\Sigma_{f2}(x,y)\phi_2(x,y))$$

$$\frac{\partial\phi_2(x,y)}{\partial t}\left(\frac{1}{v_2}\right) = \nabla\cdot D_2(x,y)\nabla\phi_2(x,y) - \Sigma_{a2}(x,y)\phi_2(x,y) + \Sigma_{s112}(x,y)\phi_2(x,y)$$

Material cross sections vary by cell and each cell has a node representing flux at its center (Figure A.8).

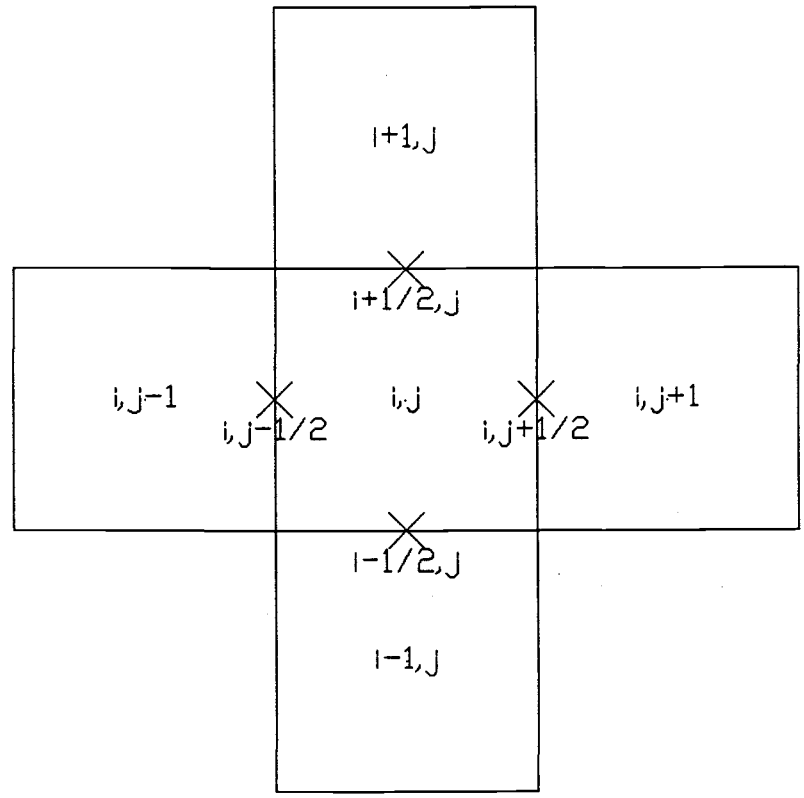

Figure A.8. Node and interfacial flux position.

Since the derivation for the thermal group equations is similar we present only the derivation for the first group equation. Integrate the equations A.2 over cell (i.e. quadrant) volume to prepare for a finite difference representation.

$$\int_V \frac{\partial \phi_1(x,y)}{\partial t}\left(\frac{1}{V_1}\right)dV = \int_V \nabla\cdot D_1(x,y)\nabla\phi_1(x,y)\,dV - \int_V \Sigma_{r1}(x,y)\phi_1(x,y) + \int_V \frac{1}{k}(\nu\Sigma_{f1}(x,y)\phi_1(x,y) + \nu\Sigma_{f2}(x,y)\phi_2(x,y))\,dV$$

Note that the divergence theorem specifies

$$\int_V \nabla\cdot D_L(x,y)\nabla\phi_L(x,y)\,dV = \int_S D_L(x,y)\nabla\phi_2(x,y)\cdot\mathbf{e}_x dS + \int_S D_L(x,y)\nabla\phi_2(x,y)\cdot\mathbf{e}_y dS$$

Vectors $\mathbf{e}_x$ and $\mathbf{e}_y$ are unit outward normals in the coordinate direction specified and originate on the surface of the cell. With this definition

$$\frac{\partial \phi_{1ij}}{\partial t}\left(\frac{V}{V_1}\right) = \int_S D_1(x,y)\nabla\phi_1(x,y)\cdot\mathbf{e}_x dS + \int_S D_1(x,y)\nabla\phi_2(x,y)\cdot\mathbf{e}_y dS + \frac{V}{k}(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)_{ij} - V\Sigma_{r1}\phi_{1ij}$$

where indices i and j are the absolute quadrant position and V is the cell volume. For the ADI method we can

implement these as

$$\frac{(\phi^* - \phi^n)_{1ij}}{\frac{\Delta t}{2}}\left(\frac{V}{V_1}\right) = \int_S D_1 \nabla \phi_1^* \cdot \mathbf{e}_x dS + \int_S D_1 \nabla \phi_1^n \cdot \mathbf{e}_y dS + \frac{V}{K}(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)_{ij}^n - V\Sigma_{r1}\phi_{1ij}^*$$

$$\frac{(\phi^{n+1} - \phi^*)_{1ij}}{\frac{\Delta t}{2}}\left(\frac{V}{V_1}\right) = \int_S D_1 \nabla \phi_1^* \cdot \mathbf{e}_x dS + \int_S D_1 \nabla \phi_1^{n+1} \cdot \mathbf{e}_y dS + \frac{V}{K}(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)_{ij}^* - V\Sigma_{r1}\phi_{1ij}^{n+1}$$

When current balance is applied at a cell interface as shown in Figure A.8 to determine the partial currents, the integral terms become

$$\int_S D_L \nabla \phi_L^* \cdot \mathbf{e}_n dS = \left[\frac{2D_{ij}D_n}{D_{ij} + D_n}\left(\frac{\phi_{ij} - \phi_n}{\Delta x}\right)\right]_L A$$

where subscript n denotes the nodal value across the interface (e.g. i,j+1), Δx is the cell width, $\mathbf{e}_n$ may be either $\mathbf{e}_x$ or $\mathbf{e}_y$, and A is the area of the interface. Since the map is only two dimensional A = Δx. Applied over the cells the equations become

$$[-Li_{ij}\phi_{ij-1}-Ui_{ij}\phi_{ij+1}+(Li+Ui+V\Sigma_r+\rho)_{ij}\phi_{ij}]_1^* =$$

$$[Lj_{ij}\phi_{i-1j}+Uj_{ij}\phi_{i+1j}-(Lj+Uj-\rho)_{ij}\phi_{ij}]_1^n+\frac{V}{k}(\nu\Sigma_{f1}\phi_1+\nu\Sigma_{f2}\phi_2)_{ij}^n$$

$$[-Li_{ij}\phi_{ij-1}-Ui_{ij}\phi_{ij+1}+(Li+Ui-\rho)_{ij}\phi_{ij}-\frac{V}{k}(\nu\Sigma_{f1}\phi_1+\nu\Sigma_{f2}\phi_2)]_1^* =$$

$$[Lj_{ij}\phi_{i-1j}+Uj_{ij}\phi_{i+1j}-(Lj+Uj+V\Sigma_r+\rho)_{ij}\phi_{ij}]_1^{n+1}$$

Similarly, the thermal group equations appear as

$$[-Li_{ij}\phi_{ij-1}-Ui_{ij}\phi_{ij+1}+(Li+Ui+V\Sigma_{a2}+\rho)_{ij}\phi_{ij}]_2^* =$$

$$[Lj_{ij}\phi_{i-1j}+Uj_{ij}\phi_{i+1j}-(Lj+Uj-\rho)_{ij}\phi_{ij}]_2^n+\Sigma_{s112}\phi_{1ij}^n$$

$$[-Li_{ij}\phi_{ij-1}-Ui_{ij}\phi_{ij+1}+(Li+Ui+V\Sigma_{a2}+\rho)_{ij}\phi_{ij}]_2^*-\Sigma_{s112}\phi_{1ij}^* =$$

$$[Lj_{ij}\phi_{i-1j}+Uj_{ij}\phi_{i+1j}-(Lj+Uj+\rho+V\Sigma_{a2})_{ij}\phi_{ij}]_2^{n+1}$$

where

$$Li_{Lij} = \left(\frac{2D_{ij}D_{ij-1}}{D_{ij}+D_{ij-1}}\right)_L, \qquad Ui_{Lij} = \left(\frac{2D_{ij}D_{ij+1}}{D_{ij}+D_{ij+1}}\right)_L$$

$$Lj_{Lij} = \left(\frac{2D_{ij}D_{i-1j}}{D_{ij}+D_{i-1j}}\right)_L, \qquad Uj_{Lij} = \left(\frac{2D_{ij}D_{i+1j}}{D_{ij}+D_{i+1j}}\right)_L$$

$$\rho_L = \frac{2V}{v_L\Delta t}$$

If we define

$$Dj1_{Lij} = -(Lj + Uj + V\Sigma_r + \rho)_{Lij}$$
$$Dj2_{Lij} = (Lj + Uj - \rho)_{Lij}$$
$$Di1_{Lij} = (Li + Ui - \rho)_{Lij}$$
$$Di2_{Lij} = -(Li + Ui + V\Sigma_r + \rho)_{Lij}$$

$$S_{1ij}^{n} = \frac{V}{k^n}(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)_{ij}^{n}$$
$$S_{2ij}^{n} = V\Sigma_{s112}\phi_{1ij}^{*}$$
$$S_{1ij}^{*} = \frac{V}{k^*}(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)_{ij}^{*}$$
$$S_{2ij}^{*} = V\Sigma_{s112}\phi_{1ij}^{n+1}$$

$$k^{n+1} = k^n \frac{\sum_i \sum_j (\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)_{ij}^{n+1}}{\sum_i \sum_j (\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)_{ij}^{n}}$$

$$k^{*} = k^{*-1} \frac{\sum_i \sum_j (\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)_{ij}^{*}}{\sum_i \sum_j (\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2)_{ij}^{*-1}}$$

The equations may now be placed in the following general form

$$[Lj_{ij}\phi_{i-1j} + Uj_{ij}\phi_{i+1j} + Dj1_{ij}\phi_{ij}]_L^{*} =$$
$$[-Li_{ij}\phi_{ij-1} - Ui_{ij}\phi_{ij+1} + Di1_{ij}\phi_{ij}]_L^{n} - S_{Lij}^{n} \qquad (A.4)$$

$$[Li_{ij}\phi_{ij-1} + Ui_{ij}\phi_{ij+1} + Di2_{ij}\phi_{ij}]_L^{n+1} =$$
$$[-Lj_{ij}\phi_{i-1j} - Uj_{ij}\phi_{i+1j} + Dj2_{ij}\phi_{ij}]_L^{*} - S_{Lij}^{*}$$

To solve the inner iteration the source terms S_{Lij}^{n} are computed and Equation A.4a is solved for each energy group by a tridiagonal matrix routine for ϕ^*. With the values of ϕ^* just computed the intermediate multiplication factor and source terms, k^* and S_{Lij}^{*} respectfully, are updated. A tridiagonal matrix routine

then solves Equation A.4b for ϕ^{n+1} of each energy group.

Note from the above definitions that the identities

$$Li_{ij} = Ui_{ij-1}$$
$$Lj_{ij} = Uj_{i-1j}$$

may be used to simplify constant computations.

Many researchers claim the key to using the ADI method for elliptic problems most efficiently lies in the proper choice of the iteration parameter ρ. Use of the ADI procedure with a fixed parameter, however, often provides a savings in computer time of 20-40% over that required by the Gauss-Seidel procedure with SOR (Anderson,1984). It is also noted that a greater savings can normally be observed if the iteration parameters are suitably varied in the ADI procedure (Carnahan,1969;Gladwell,1979).

The method as applied here uses $\rho = 0$ for simplicity. There is no reason to believe that such a value is not sufficient since, after evaluating a few loading patterns for convergence on flux, only at most 4 time step iterations on flux were required for convergence to less than .01% upon some particularly tough convergence nodes in the map (e.g. those nodes nearest the reflecting boundaries). The method is extremely strong on convergence and no value of ρ other than 0 or more than a single inner iteration appears necessary.

A.5.5 Boundary Conditions

Reflecting boundary conditions are imposed at the top and left symmetrical axis of the map and require the derivative of the flux on the boundary to be zero. This requires $\phi_{i0} = \phi_{i1}$ and $\phi_{0j} = \phi_{1j}$. These conditions are ensured by setting the coefficients multiplying the flux terms to zero. That is

$$Li_{i1} = 0$$
$$Lj_{1j} = 0$$

Extrapolated boundary conditions are applied to peripheral positions in the map. From these nodes the flux is assumed to extrapolate to zero at the distance

$$l_L = \frac{\Delta x}{2} + d_L$$

into the moderator where d_L is the extrapolation distance in the moderator for energy group L. At the extra nodes in the moderator at the map periphery the constants change for extrapolation as

$$Ui_{Lij} = \left(\frac{D_{ij}\Delta x}{1}\right)_{L,\ j\ @\ \text{boundary}}$$

$$Uj_{Lij} = \left(\frac{D_{ij}\Delta x}{1}\right)_{L,\ i\ @\ \text{boundary}}$$

A.5.6 SOR Outer Iteration

For each ADI inner iteration the total fissions are computed and the multiplication factor k^{n+1} is determined. When the total fissions between outer iterations differ by less that a convergence criteria the problem has converged. To accelerate convergence of the problem, the fissions at each position are overrelaxed by the equation

$$s_{ij}^{n+1} = s_{ij}^{n} + \alpha\left(\sum_{i}\sum_{j}\left(\nu\Sigma_{f1}\phi_1 + \nu\Sigma_{f2}\phi_2\right)_{ij} - s_{ij}^{n}\right)$$

where α is the acceleration factor and s_{ij}^{n} was computed from a prior iteration.

A.5.8 Flux averaging

Accuracy of coarse mesh nodal calculations suffer due to the large separation between comparatively few nodes. The number of nodes needed to represent flux gradients correctly is subject to some experimentation and changes with the magnitude of the gradients. In an attempt to improve the power calculations, once the outer iteration converges flux averaging is imposed at each node by balancing the nodal flux with the fluxes on the cell interface. An expression for the flux at the cell interface may be determined by applying current

continuity at the interface (Figure A.8). For example, the flux at the i, j+1/2 position is

$$\phi_{ij+\frac{1}{2}} = \frac{\phi_{ij}D_{ij} + \phi_{ij+1}D_{ij}}{D_{ij+1} + D_{ij+1}}$$

Fluxes at the remaining three interfaces are computed similarly. The average nodal flux is then

$$\bar{\phi}_{Lij} = A_L\phi_{Lij} + \frac{1-A_L}{4}(\phi_{ij+\frac{1}{2}} + \phi_{i+\frac{1}{2}j} + \phi_{i-\frac{1}{2}j} + \phi_{ij-\frac{1}{2}})$$

where A_L is the Borresen weighting factor for energy group L.

A.5.9 Power and Burnup Calculation

Once fluxes have been adjusted, the relative nodal power is given by

$$P_{ij} = \frac{(\nu\Sigma_{f1}\bar{\phi}_1 + \nu\Sigma_{f2}\bar{\phi}_2)}{\bar{P}}$$

where

$$\bar{P} = \frac{\sum_i\sum_j (\nu\Sigma_{f1}\bar{\phi}_1 + \nu\Sigma_{f2}\bar{\phi}_2)_{ij}}{n_{fuel}}$$

and n_{fuel} is the total number of fueled nodes. When a burn calculation is performed, exposures of all nodes are increased in proportion to their average power such that

$$X_{ij}^{n+1} = X_{ij}^{n} + \Delta XP_{ij}$$

where X_{ij}^{n+1} is the node exposure following map burnup step ΔX and X_{ij}^{n} is the prior node exposure. Cross sections are then recalculated for the new exposure and another power calculation is completed for the next burn step. Care must be taken to limit the burnup ΔX to ensure solution accuracy.

A.6 Verification

To evaluate PWRCALC the fine mesh nodal code 2DB was used. Both programs were run with a simplified cross section set representative of Trojan's Cycle 6 loading if all bundles had zero exposure and boron is not present. It was found the minimum difference in power between the calculations could be obtained if the Borresen weighting factors were set to zero for maximum flux averaging. Figure A.9 shows the difference between calculations is at worst 5%.

PWRCALC was evaluated in terms of the Trojan Cycle 10 initial loading as proposed by PGE (Figure A.10) and the difference by average bundle power between the calculations is shown in Figure A.11. The input file constructed for this test is shown in Figure A.12 and Figures A.13 and A.14 show the corresponding screen and BUNDLES.PWR output generated. Finally, Figure A.15

```
1.635 1.678 1.656 1.608 1.578 1.534 1.492 1.453 1.400 1.311 1.182  .998  .805  .600  .458
1.611 1.631 1.612 1.583 1.553 1.502 1.464 1.438 1.390 1.305 1.180 1.005  .815  .608  .440
 .024  .047  .044  .026  .025  .032  .028  .015  .010  .006  .002 -.007 -.010 -.008  .019

1.678 1.312 1.299 1.595 1.561 1.221 1.188 1.439 1.361 1.275 1.147 1.003  .806  .589  .451
1.631 1.328 1.314 1.558 1.527 1.239 1.207 1.414 1.356 1.274 1.151 1.006  .814  .599  .433
 .047 -.016 -.015  .037  .034 -.018 -.019  .025  .005  .001 -.003 -.003 -.008 -.010  .018

1.656 1.299 1.285 1.574 1.540 1.204 1.170 1.413 1.334 1.248 1.121  .974  .781  .570  .434
1.612 1.314 1.300 1.538 1.507 1.222 1.189 1.390 1.330 1.248 1.124  .980  .792  .581  .419
 .044 -.015 -.015  .036  .032 -.019 -.019  .023  .004  .001 -.004 -.006 -.011 -.011  .016

1.608 1.595 1.574 1.551 1.516 1.470 1.423 1.377 1.319 1.207 1.079  .933  .743  .539  .408
1.583 1.558 1.538 1.528 1.495 1.442 1.399 1.367 1.313 1.209 1.085  .941  .756  .552  .396
 .026  .037  .036  .023  .021  .028  .024  .010  .006 -.002 -.006 -.008 -.013 -.013  .012

1.577 1.561 1.539 1.516 1.479 1.429 1.379 1.330 1.268 1.151 1.022  .877  .690  .493  .367
1.553 1.527 1.507 1.495 1.461 1.406 1.359 1.323 1.264 1.157 1.032  .888  .706  .509  .361
 .025  .034  .032  .020  .018  .024  .020  .007  .003 -.006 -.010 -.011 -.016 -.017  .006

1.534 1.220 1.203 1.470 1.429 1.108 1.065 1.271 1.179 1.102  .967  .802  .618  .438  .313
1.502 1.239 1.222 1.442 1.406 1.131 1.089 1.257 1.185 1.108  .977  .818  .637  .455  .314
 .032 -.019 -.019  .028  .024 -.023 -.024  .014 -.006 -.006 -.011 -.016 -.020 -.017 -.002

1.492 1.188 1.169 1.423 1.378 1.065 1.018 1.203 1.105 1.018  .879  .713  .533  .464  .273
1.463 1.207 1.189 1.399 1.359 1.089 1.042 1.193 1.114 1.029  .895  .733  .558  .436  .288
 .028 -.019 -.020  .024  .020 -.024 -.024  .009 -.009 -.011 -.016 -.019 -.025  .028 -.016

1.452 1.438 1.412 1.376 1.329 1.271 1.202 1.126 1.039  .924  .782  .616  .542
1.438 1.414 1.390 1.367 1.322 1.257 1.193 1.131 1.049  .939  .801  .636  .520
 .014  .024  .023  .010  .007  .013  .009 -.005 -.010 -.015 -.020 -.021  .022

1.399 1.360 1.333 1.318 1.267 1.179 1.105 1.039  .938  .817  .674  .511  .424
1.389 1.356 1.330 1.313 1.264 1.185 1.114 1.049  .953  .836  .696  .534  .421
 .010  .005  .003  .006  .003 -.006 -.009 -.010 -.014 -.018 -.022 -.023  .003

1.309 1.274 1.247 1.206 1.150 1.101 1.018  .924  .817  .694  .554  .412  .318
1.305 1.274 1.247 1.208 1.157 1.107 1.029  .939  .836  .716  .579  .434  .322
 .005  .000  .000 -.003 -.006 -.006 -.011 -.015 -.018 -.021 -.024 -.022 -.005

1.181 1.146 1.119 1.078 1.021  .966  .879  .781  .673  .554  .427  .376  .239
1.180 1.150 1.124 1.085 1.031  .977  .895  .801  .696  .579  .455  .369  .260
 .001 -.004 -.005 -.007 -.010 -.011 -.016 -.020 -.022 -.024 -.028  .007 -.021

 .997 1.001  .973  .932  .876  .801  .713  .615  .511  .412  .376
1.005 1.006  .979  .941  .887  .818  .733  .636  .534  .434  .369
-.007 -.004 -.007 -.009 -.012 -.016 -.020 -.021 -.023 -.022  .007

 .804  .805  .780  .742  .689  .617  .532  .541  .424  .317  .239
 .814  .814  .791  .756  .706  .637  .558  .520  .421  .322  .260
-.011 -.009 -.011 -.014 -.017 -.020 -.025  .021  .003 -.005 -.021

 .599  .588  .569  .538  .492  .437  .464
 .608  .599  .581  .552  .509  .455  .436
-.009 -.010 -.012 -.014 -.017 -.017  .028

 .457  .450  .434  .407  .366  .312  .272
 .439  .433  .419  .396  .361  .314  .288
 .018  .017  .015  .012  .006 -.002 -.016

1 -- PWRCALC
2 -- 2DB
3 -- PWRCALC - 2DB
```

Figure A.9. Deviation of 2DB and PWRCALC power predictions for loading representative of Cycle 6.

J=	R 1	P 2	N 3	M 4	L 5	K 6	J 7	H 8	G 9	F 10	E 11	D 12	C 13	B 14	A 15
I 1					0.401 0.738 22.440	0.443 0.744 32.159	0.480 0.759 32.102	0.712 0.959 0.0	0.480 0.758 32.168	0.441 0.740 32.155	0.401---AVE. POWER 0.737---PEAK POWER 22.406---AVE. EXPOSURE				CYCLE EXP = 0.0 CORE AVE EXP = 16.6107 AVE ASSM WT = 461.885 BORON PPM = 1108.0 LAMBDA = 0.999291
2			0.349 0.728 32.311	0.942 1.279 0.0	1.162 1.359 0.0	1.219 1.383 0.0	1.196 1.357 0.0	0.996 1.083 20.726	1.196 1.356 0.0	1.216 1.378 0.0	1.159 1.354 0.0	0.943 1.279 0.0	0.350 0.725 32.304		
3		0.351 0.726 32.261	0.784 1.085 20.686	1.258 1.383 0.0	0.938 0.993 31.986	0.961 1.027 24.983	0.875 0.925 30.545	1.243 1.357 0.0	0.880 0.931 30.507	0.953 1.023 25.010	0.933 0.991 32.010	1.260 1.385 0.0	0.784 1.084 20.695	0.349 0.728 32.350	
4		0.943 1.279 0.0	1.260 1.385 0.0	0.995 1.053 30.483	1.212 1.322 11.894	1.253 1.362 7.192	0.965 1.067 24.907	0.928 0.996 29.073	1.003 1.113 23.202	1.215 1.305 9.666	1.202 1.306 11.887	0.994 1.053 30.485	1.257 1.382 0.0	0.942 1.278 0.0	
5	0.400 0.736 22.418	1.159 1.352 0.0	0.932 0.991 32.040	1.202 1.306 11.890	1.284 1.362 11.800	1.145 1.197 25.074	1.274 1.379 12.144	1.266 1.380 11.774	1.284 1.383 12.181	1.166 1.223 23.231	1.284 1.362 11.800	1.212 1.321 11.903	0.937 0.992 31.986	1.161 1.358 0.0	0.401 0.736 22.463
6	0.441 0.740 32.136	1.215 1.378 0.0	0.952 1.023 25.010	1.215 1.302 9.672	1.165 1.222 23.259	1.327 1.414 12.769	1.333 1.411 12.539	1.197 1.271 23.177	1.333 1.414 12.523	1.327 1.414 12.759	1.145 1.197 25.069	1.252 1.362 7.186	0.960 1.026 24.986	1.218 1.382 0.0	0.442 0.742 32.151
7	0.480 0.759 32.115	1.195 1.356 0.0	0.879 0.930 30.557	1.003 1.112 23.208	1.284 1.383 12.179	1.333 1.415 12.538	1.162 1.227 25.077	1.288 1.371 9.660	1.161 1.226 25.065	1.333 1.411 12.539	1.274 1.379 12.147	0.964 1.067 24.883	0.874 0.924 30.585	1.194 1.355 0.0	0.479 0.755 32.179
8	0.712 0.959 0.0	0.996 1.083 20.729	1.243 1.357 0.0	0.929 0.997 29.045	1.266 1.381 11.777	1.198 1.273 23.168	1.291 1.375 9.658	1.022 1.071 30.575	1.288 1.370 9.641	1.197 1.272 23.166	1.266 1.381 11.741	0.928 0.996 29.064	1.241 1.355 0.0	0.995 1.082 20.727	0.711 0.958 0.0
9	0.480 0.756 32.125	1.195 1.357 0.0	0.875 0.925 30.563	0.965 1.067 24.926	1.275 1.381 12.103	1.334 1.412 12.531	1.163 1.228 25.061	1.291 1.374 9.660	1.161 1.227 25.069	1.332 1.414 12.538	1.284 1.382 12.174	1.002 1.115 23.209	0.879 0.929 30.551	1.194 1.355 0.0	0.479 0.754 32.169
10	0.443 0.744 32.101	1.219 1.383 0.0	0.961 1.027 24.988	1.253 1.366 7.181	1.145 1.197 25.105	1.328 1.415 12.760	1.334 1.415 12.536	1.198 1.272 23.171	1.333 1.411 12.541	1.327 1.414 12.767	1.165 1.222 23.234	1.214 1.304 9.670	0.952 1.022 25.009	1.215 1.377 0.0	0.441 0.739 32.178
11	0.401 0.738 22.454	1.162 1.358 0.0	0.938 0.993 31.989	1.212 1.321 11.905	1.284 1.362 11.805	1.166 1.223 23.233	1.264 1.383 12.184	1.266 1.381 11.776	1.284 1.379 12.142	1.145 1.197 25.070	1.283 1.361 11.806	1.201 1.305 11.887	0.942 0.990 32.034	1.158 1.353 0.0	0.400 0.736 22.364
12		0.942 1.278 0.0	1.257 1.383 0.0	0.995 1.053 30.482	1.202 1.306 11.869	1.215 1.305 9.671	1.003 1.113 23.210	0.929 0.997 29.056	0.965 1.067 24.887	1.252 1.362 7.191	1.212 1.321 11.891	0.994 1.052 30.494	1.260 1.384 0.0	0.943 1.279 0.0	
13		0.349 0.728 32.338	0.784 1.085 20.695	1.260 1.385 0.0	0.933 0.991 32.038	0.953 1.023 25.003	0.880 0.931 30.501	1.243 1.357 0.0	0.875 0.925 30.539	0.960 1.027 24.985	0.938 0.993 31.976	1.257 1.382 0.0	0.784 1.084 20.689	0.351 0.726 32.246	
14			0.351 0.726 32.251	0.944 1.279 0.0	1.159 1.354 0.0	1.216 1.378 0.0	1.196 1.356 0.0	0.996 1.083 20.728	1.195 1.357 0.0	1.219 1.383 0.0	1.161 1.358 0.0	0.942 1.278 0.0	0.349 0.726 32.333		
15					0.401 0.737 22.401	0.441 0.740 32.149	0.480 0.759 32.143	0.712 0.959 0.0	0.480 0.756 32.122	0.443 0.743 32.147	0.401 0.738 22.478				

-------POWER------		-TEMPERATURES(DEGREE F)-		
THERMAL	RELATIVE	AVERAGE	INLET	EXIT
3411.0	1.00000	584.6	553.8	615.3

Figure A.10. PGE Cycle 10 loading and bundle powers.

1.022	1.288	1.197	1.266	.928	1.241	.995	.711
.921	1.247	1.143	1.245	.900	1.300	1.060	.708
.101	.041	.054	.021	.028	-.059	-.065	.003
1.291	1.161	1.331	1.284	1.002	.879	1.194	.479
1.244	1.097	1.302	1.255	.970	.871	1.253	.488
.047	.064	.029	.029	.032	.008	-.059	-.009
1.198	1.333	1.327	1.165	1.214	.952	1.215	.441
1.131	1.291	1.300	1.128	1.204	.956	1.270	.443
.067	.042	.027	.037	.010	-.004	-.055	-.002
1.266	1.274	1.145	1.283	1.201	.932	1.158	.400
1.217	1.231	1.101	1.285	1.217	.932	1.187	.424
.049	.043	.044	-.002	-.016	.000	-.029	-.024
.929	.965	1.252	1.212	.994	1.260	.943	
.874	.916	1.248	1.232	.986	1.283	.986	
.055	.049	.004	-.020	.008	-.023	-.043	
1.243	.875	.960	.938	1.257	.784	.351	
1.278	.863	.967	.941	1.288	.779	.354	
-.035	.012	-.007	-.003	-.031	.005	-.003	
.996	1.185	1.219	1.161	.942	.349		
1.052	1.249	1.275	1.195	.990	.355		
-.056	-.064	-.056	-.034	-.048	-.006		
.712	.480	.443	.401				
.706	.487	.444	.425				
.006	-.007	-.001	-.024				

1 -- PGE
2 -- PWRCALC
3 -- PGE - PWRCALC

Figure A.11. Power deviation of PGE predicted and PWRCALC results for Cycle 10 initial loading.

```
        56      1108.000000            0           0           0
 1  1 'D53'  2  1 30.57500 30.57500 30.57500 30.57500
 2  1 'L02'  6  1  9.64100  9.64100  9.64100  9.64100
 3  1 'K05'  6  1 23.16600 23.16600 23.16600 23.16600
 4  1 'K03'  6  1 11.74100 11.74100 11.74100 11.74100
 5  1 'H43'  4  1 29.06400 29.06400 29.06400 29.06400
 6  1 'M47'  5  1   .00000   .00000   .00000   .00000
 7  1 'K11'  6  1 20.72700 20.72700 20.72700 20.72700
 8  1 'M14'  5  1   .00000   .00000   .00000   .00000
 1  2 'L28'  6  1  9.66000  9.66000  9.66000  9.66000
 2  2 'K36'  6  1 25.06900 25.06900 25.06900 25.06900
 3  2 'L08'  6  1 12.53800 12.53800 12.53800 12.53800
 4  2 'L14'  6  1 12.17400 12.17400 12.17400 12.17400
 5  2 'K25'  6  1 23.20900 23.20900 23.20900 23.20900
 6  2 'C35'  1  1 30.55100 30.55100 30.55100 30.55100
 7  2 'M02'  5  1   .00000   .00000   .00000   .00000
 8  2 'H33'  4  1 32.16900 32.16900 32.16900 32.16900
 1  3 'K28'  6  1 23.17100 23.17100 23.17100 23.17100
 2  3 'L15'  6  1 12.54100 12.54100 12.54100 12.54100
 3  3 'L17'  6  1 12.76700 12.76700 12.76700 12.76700
 4  3 'K44'  6  1 23.23400 23.23400 23.23400 23.23400
 5  3 'L30'  6  1  9.67000  9.67000  9.67000  9.67000
 6  3 'J10'  4  1 25.00900 25.00900 25.00900 25.00900
 7  3 'M35'  5  1   .00000   .00000   .00000   .00000
 8  3 'H38'  4  1 32.17800 32.17800 32.17800 32.17800
 1  4 'K29'  6  1 11.77600 11.77600 11.77600 11.77600
 2  4 'L04'  6  1 12.14200 12.14200 12.14200 12.14200
 3  4 'K37'  6  1 25.07000 25.07000 25.07000 25.07000
 4  4 'K08'  6  1 11.80600 11.80600 11.80600 11.80600
 5  4 'K17'  6  1 11.88700 11.88700 11.88700 11.88700
 6  4 'H10'  4  1 32.03400 32.03400 32.03400 32.03400
 7  4 'M41'  5  1   .00000   .00000   .00000   .00000
 8  4 'G03'  3  1 22.36400 22.36400 22.36400 22.36400
 1  5 'H05'  4  1 29.05600 29.05600 29.05600 29.05600
 2  5 'J02'  4  1 24.88700 24.88700 24.88700 24.88700
 3  5 'L01'  6  1  7.19100  7.19100  7.19100  7.19100
 4  5 'K14'  6  1 11.89100 11.89100 11.89100 11.89100
 5  5 'H08'  4  1 30.49400 30.49400 30.49400 30.49400
 6  5 'M32'  5  1   .00000   .00000   .00000   .00000
 7  5 'M24'  5  1   .00000   .00000   .00000   .00000
 1  6 'M45'  5  1   .00000   .00000   .00000   .00000
 2  6 'C63'  1  1 30.53900 30.53900 30.53900 30.53900
 3  6 'J04'  4  1 24.98500 24.98500 24.98500 24.98500
 4  6 'H41'  4  1 31.97600 31.97600 31.97600 31.97600
 5  6 'M23'  5  1   .00000   .00000   .00000   .00000
 6  6 'K10'  6  1 20.68900 20.68900 20.68900 20.68900
 7  6 'H31'  4  1 32.24600 32.24600 32.24600 32.24600
 1  7 'K04'  6  1 20.72800 20.72800 20.72800 20.72800
 2  7 'M10'  5  1   .00000   .00000   .00000   .00000
 3  7 'M30'  5  1   .00000   .00000   .00000   .00000
 4  7 'M21'  5  1   .00000   .00000   .00000   .00000
 5  7 'M09'  5  1   .00000   .00000   .00000   .00000
 6  7 'H13'  4  1 32.33300 32.33300 32.33300 32.33300
 1  8 'M36'  5  1   .00000   .00000   .00000   .00000
 2  8 'H39'  4  1 32.12200 32.12200 32.12200 32.12200
 3  8 'H40'  4  1 32.14700 32.14700 32.14700 32.14700
 4  8 'G10'  3  1 22.47800 22.47800 22.47800 22.47800
```

Figure A.12. PWRCALC sample input (BUNDLES.DAT) for cycle 10 initial loading.

ITOUT	K-EFF	K-EFF*	PHI1(1,1)	PHI1(4,4)	FISSIONS	CONVERGENCE	LAMBDA	LAMBDA*
1	.98826	1.01280	.90655	1.07002	.5886E+03	.1188E-01	.988259	1.012805
4	1.01439	1.01392	.91765	1.16827	.6042E+03	.9916E-03	1.000993	1.001452
7	1.01510	1.01492	.92482	1.20139	.6046E+03	.1678E-03	1.000168	1.000094
10	1.01531	1.01517	.95461	1.25369	.6048E+03	.3623E-04	1.000036	1.000067
13	1.01537	1.01530	.99905	1.29701	.6048E+03	.1403E-04	1.000014	1.000035
16	1.01538	1.01536	1.03256	1.31722	.6048E+03	.3532E-05	1.000004	1.000014
19	1.01539	1.01538	1.04199	1.32274	.6048E+03	.1110E-05	1.000001	1.000007
22	1.01539	1.01540	1.03920	1.32202	.6048E+03	.9082E-06	1.000001	1.000003

***** RELATIVE POWER FOR I,J POSITION *****

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	.92	1.23	1.26	1.14	1.15	1.28	1.21	.90	.90	1.27	1.34	1.14	.99	.84	.57
2	1.23	1.07	1.10	1.27	1.30	1.30	1.21	.98	.93	.85	.94	1.31	1.16	.63	.39
3	1.26	1.09	1.12	1.31	1.33	1.29	1.22	1.02	.95	.81	.89	1.35	1.20	.60	.34
4	1.13	1.26	1.30	1.32	1.31	1.14	1.11	1.21	1.14	.92	.98	1.36	1.20	.58	.32
5	1.13	1.29	1.31	1.30	1.27	1.14	1.13	1.26	1.20	.94	.99	1.35	1.18	.56	.31
6	1.26	1.27	1.27	1.11	1.11	1.27	1.29	1.27	1.19	.89	.92	1.31	1.13	.56	.30
7	1.18	1.18	1.19	1.08	1.11	1.29	1.29	1.24	1.18	.94	.97	1.27	1.03	.58	.27
8	.87	.92	.97	1.24	1.31	1.28	1.25	.98	.98	1.32	1.33	1.22	.99		
9	.87	.87	.90	1.18	1.25	1.21	1.19	.99	1.00	1.28	1.21	1.00	.74		
10	1.24	.84	.80	.93	.96	.90	.95	1.32	1.28	.99	.79	.51	.35		
11	1.32	.93	.88	.99	1.00	.93	.98	1.33	1.21	.79	.54	.35	.20		
12	1.12	1.30	1.34	1.36	1.35	1.32	1.28	1.23	1.00	.51	.35				
13	.98	1.16	1.20	1.20	1.18	1.14	1.04	.99	.74	.35	.20				
14	.84	.63	.59	.58	.57	.56	.58								
15	.57	.38	.34	.32	.31	.30	.27								
16															
17															

Stop - Program terminated.

Figure A.13. PWRCALC screen output for Cycle 10 initial loading.

```
D53   1  1  2  1  30.57500 30.57500 30.57500 30.57500  .87134  .87134  .87134  .87134  .91935  .91935  .91935  .91935
L02   2  1  6  1   9.64100  9.64100  9.64100  9.64100 1.08632 1.08632 1.08632 1.08632 1.22752 1.26469 1.22752 1.26469
K05   3  1  6  1  23.16600 23.16600 23.16600 23.16600  .95806  .95806  .95806  .95806 1.13542 1.14965 1.13542 1.14965
K03   4  1  6  1  11.74100 11.74100 11.74100 11.74100 1.06438 1.06438 1.06438 1.06438 1.28235 1.20784 1.28235 1.20784
H43   5  1  4  1  29.06400 29.06400 29.06400 29.06400  .90013  .90013  .90013  .90013  .90084  .90042  .90084  .90042
M47   6  1  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.26693 1.33517 1.26693 1.33517
K11   7  1  6  1  20.72700 20.72700 20.72700 20.72700  .97893  .97893  .97893  .97893 1.13514  .98555 1.13514  .98555
M14   8  1  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440  .84496  .57237  .84496  .57237
L28   1  2  6  1   9.66000  9.66000  9.66000  9.66000 1.08612 1.08612 1.08612 1.08612 1.22526 1.22526 1.25877 1.25877
K36   2  2  6  1  25.06900 25.06900 25.06900 25.06900  .94243  .94243  .94243  .94243 1.07300 1.09779 1.09440 1.11736
L08   3  2  6  1  12.53800 12.53800 12.53800 12.53800 1.05624 1.05624 1.05624 1.05624 1.27238 1.30352 1.30588 1.32582
L14   4  2  6  1  12.17400 12.17400 12.17400 12.17400 1.05994 1.05994 1.05994 1.05994 1.29893 1.21401 1.29135 1.21657
K25   5  2  6  1  23.20900 23.20900 23.20900 23.20900  .95770  .95770  .95770  .95770  .98352  .92929 1.02072  .94925
C35   6  2  1  1  30.55100 30.55100 30.55100 30.55100  .87063  .87063  .87063  .87063  .85482  .93736  .80650  .88761
M02   7  2  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.30804 1.16080 1.34536 1.19945
H33   8  2  4  1  32.16900 32.16900 32.16900 32.16900  .87737  .87737  .87737  .87737  .62902  .38545  .59507  .34177
K28   1  3  6  1  23.17100 23.17100 23.17100 23.17100  .95802  .95802  .95802  .95802 1.12564 1.12564 1.13365 1.13365
L15   2  3  6  1  12.54100 12.54100 12.54100 12.54100 1.05621 1.05621 1.05621 1.05621 1.26303 1.29984 1.28684 1.31156
L17   3  3  6  1  12.76700 12.76700 12.76700 12.76700 1.05393 1.05393 1.05393 1.05393 1.32394 1.30616 1.29674 1.27184
K44   4  3  6  1  23.23400 23.23400 23.23400 23.23400  .95749  .95749  .95749  .95749 1.14189 1.10632 1.13528 1.12787
L30   5  3  6  1   9.67000  9.67000  9.67000  9.67000 1.08601 1.08601 1.08601 1.08601 1.21178 1.14362 1.26500 1.19754
J10   6  3  4  1  25.00900 25.00900 25.00900 25.00900  .93168  .93168  .93168  .93168  .91665  .98138  .94014  .98648
M35   7  3  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.35524 1.20124 1.34625 1.18024
H38   8  3  4  1  32.17800 32.17800 32.17800 32.17800  .87730  .87730  .87730  .87730  .57999  .32244  .56344  .30714
K29   1  4  6  1  11.77600 11.77600 11.77600 11.77600 1.06402 1.06402 1.06402 1.06402 1.25646 1.25646 1.17551 1.17551
L04   2  4  6  1  12.14200 12.14200 12.14200 12.14200 1.06027 1.06027 1.06027 1.06027 1.27424 1.26906 1.18408 1.19344
K37   3  4  6  1  25.07000 25.07000 25.07000 25.07000  .94242  .94242  .94242  .94242 1.10676 1.10815 1.07911 1.10997
K08   4  4  6  1  11.80600 11.80600 11.80600 11.80600 1.06371 1.06371 1.06371 1.06371 1.27348 1.28656 1.28947 1.29067
K17   5  4  6  1  11.88700 11.88700 11.88700 11.88700 1.06288 1.06288 1.06288 1.06288 1.26642 1.18713 1.23829 1.17873
H10   8  4  4  1  32.03400 32.03400 32.03400 32.03400  .87834  .87834  .87834  .87834  .88990  .92367  .94415  .96967
M41   7  4  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.31450 1.12962 1.27382 1.03340
G03   8  4  3  1  22.36400 22.36400 22.36400 22.36400  .93636  .93636  .93636  .93636  .55928  .29619  .57515  .26510
H05   1  5  4  1  29.05600 29.05600 29.05600 29.05600  .90019  .90019  .90019  .90019  .87293  .87293  .87435  .87435
J02   2  5  4  1  24.68700 24.68700 24.68700 24.68700  .93267  .93267  .93267  .93267  .91819  .96838  .86948  .90483
L01   3  5  6  1   7.19100  7.19100  7.19100  7.19100 1.11288 1.11288 1.11288 1.11288 1.24493 1.31180 1.18376 1.25086
K14   4  5  6  1  11.89100 11.89100 11.89100 11.89100 1.06284 1.06284 1.06284 1.06284 1.28451 1.24529 1.20987 1.18945
H08   5  5  4  1  30.49400 30.49400 30.49400 30.49400  .88952  .88952  .88952  .88952  .97780  .98287  .98641  .99948
M32   6  5  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.31702 1.32690 1.27847 1.21016
M24   7  5  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.21880  .98773  .99951  .73786
M45   1  6  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.23973 1.23973 1.31636 1.31636
C83   2  6  1  1  30.53900 30.53900 30.53900 30.53900  .87071  .87071  .87071  .87071  .83899  .80077  .92699  .88385
J04   3  6  4  1  24.98500 24.98500 24.98500 24.98500  .93187  .93187  .93187  .93187  .92705  .95829  .98582  .99560
H41   4  6  4  1  31.97600 31.97600 31.97600 31.97600  .87875  .87875  .87875  .87875  .90259  .95276  .93251  .97714
M23   5  6  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.32338 1.28133 1.33369 1.21422
K10   8  8  8  1  20.68900 20.68900 20.68900 20.68900  .97926  .97926  .97926  .97926  .98670  .79109  .79238  .54467
H31   7  8  4  1  32.24600 32.24600 32.24600 32.24600  .87681  .87681  .87681  .87681  .51089  .35120  .35057  .20403
K04   1  7  8  1  20.72800 20.72800 20.72800 20.72800  .97892  .97892  .97892  .97892 1.12495 1.12495  .97967  .97967
M10   2  7  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.29974 1.34231 1.15584 1.19771
M30   3  7  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.35829 1.35341 1.20325 1.18496
M21   4  7  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.32285 1.28151 1.13537 1.03885
M09   5  7  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440 1.22501 1.00342  .99253  .74089
H13   8  7  4  1  32.33300 32.33300 32.33300 32.33300  .87619  .87619  .87619  .87619  .51177  .35060  .35198  .20419
M36   1  8  5  1   .00000  .00000  .00000  .00000 1.19440 1.19440 1.19440 1.19440  .84119  .84119  .57027  .57027
H39   2  8  4  1  32.12200 32.12200 32.12200 32.12200  .87770  .87770  .87770  .87770  .62724  .59474  .38462  .34170
H40   3  8  4  1  32.14700 32.14700 32.14700 32.14700  .87752  .87752  .87752  .87752  .58102  .56548  .32299  .30810
G10   4  8  3  1  22.47800 22.47800 22.47800 22.47800  .93539  .93539  .93539  .93539  .56101  .57710  .29687  .26580
 1.01539  1108.00   .0000  3  2
```

Figure A.14. PWRCALC sample output (BUNDLES.PWR) for Cycle 10 initial loading.

```
      PROGRAM PWRCLC
      COMMON/NUCLER/sigfnu(0:20,0:20,2),SIGS12(0:20,0:20)
     $,siga(0:20,0:20,2),D(0:20,0:20,2)
      COMMON/CXDATA/CXA1(8,8,5),CXA2(8,8,5),CXF1(8,8,5),CXF2(8,8,5),
     $             CXD1(8,8,5),CXD2(8,8,5),CXSC(8,8,5),PPMB
      COMMON/DIMEN/NXA,NYA,LNXF(0:20),LNX(0:20),LNY(0:20),AREA,TFUEL,
     $            CC(2),EXT(2),lnxb(0:8),maxl,maxk
      COMMON/ADI/EFFKS,TF(20,20,2),EFFK,RLAMS,TLFSLS,UK(20,20,2)
     $ ,LK(20,20,2),DK2(20,20,2),UJ(20,20,2),LJ(20,20,2),DJ1(20,20,2)
     $,DJ2(20,20,2),DK1(20,20,2)
      DIMENSION EX(0:20,0:20),RE(0:20,0:20),index(5),rpow(0:20,0:20),
     $      idx(8,8),rcxa2(5),rcxf1(5),rcxf2(5),rcxsc(5),rcxd1(5),
     $      rcxa1(5),rcxd2(5),LNYB(0:8),irot(8,8)
      logical read,search
      CHARACTER BID(8,8)*5,char
      data nuniq /1/
C
C     ***** DESCRIPTION OF VARIABLES *****
C
C  NXA       - NUMBER OF X OR Y NODES FROM CENTER TO CORE BOUNDARY (17 RECOMMENDED)
C  MAXO      - MAXIMUM NUMBER OF OUTER ( K  ) ITERATIONS
C  SIZE      - NODE WIDTH (CM)
C  TESTOT    - CONVERGENCE CRITERIA ON OUTER ( K  ) ITERATIONS (1E-6 RECOMMENDED)
C  AA(1)     - FAST BORRENSON AVERAGING FACTOR
C  AA(2)     - THERMAL BORRESON AVERAGING FACTOR
C  ALPHAO    - OVERRELAXATION FACTOR ON OUTER ITERATIONS (1.7 RECOMMENDED)
C  TFUEL     - TOTAL NUMBER OF FUELED NODES IN QUARTER CORE
C  PPMB      - NATURAL BORON CONCENTRATION (PPM)
C  XNHYD     - VOLUME WEIGHTED NUMBER DENSITY OF HYDROGEN FROM LEOPARD (LONG PRINT)
C  CXBMIC    - B-10 AVERAGE THERMAL MICRO X-SECTION @ MODERATOR TEMP FROM LEOPARD (LONG PRINT)
C  H2ODEN    - WATER DENSITY AT OPERATING PRESSURE AND TEMPERATURE (OPAT) FROM LEOPARD
C  DWATER(1) - FAST DIFFUSION COEFFICENT OF WATER AT OPAT FROM LEOPARD
C  DWATER(2) - THERMAL DIFFUSION COEFFICENT OF WATER AT OPAT FROM LEOPARD
C  SIGAW(1)  - FAST MACRO ABSORPTION COEFFICENT OF WATER AT OPAT FROM LEOPARD
C  SIGAW(2)  - THERMAL MACRO ABSORPTION COEFFICENT OF WATER AT OPAT FROM LEOPARD
C  SIGSLW    - SLOWING DOWN CROSS SECTION FROM LEOPARD
C  LNX(I)    - NUMBER OF NODES REPRESENTED PER ROW FROM CENTER.  NORMALLY LNXF(I)+2
C  LNXF(I)   - NUMBER OF FUELED NODES REPRESENTED PER ROW FROM CENTER.
C  LNXB(I)   - TOTAL NUMBER OF BUNDLES IN ROW I.
C  LNYB(I)   - TOTAL NUMBER OF BUNDLES IN COLUMN I.
C  EX(I,J)   - ARRAY OF FUEL EXPOSURES IN QUARTER CORE.
C  RPOW(I,J) - RELATIVE POWER BY LOADING POSITION.
C  RE(I,J)   - REACTIVITY BY LOADING POSITION.
C  IDX(L,K)  - CROSS SECTION SET INDEX BY BUNDLE.
C  IROT(L,K) - BUNDLE ORIENTATION
C  EXPI      - INITIAL EXPOSURE OF LOADING (GWD/MT).
C  EXPF      - FINAL EXPOSURE OF LOADING.
C  EXPINC    - EXPOSURE INCREMENT.
C  EFFK      - LOADING REACTIVITY.
C
C  COEFFICENTS FOR A FOURTH ORDER CORRELATION OF CROSS SECTION TO EXPOSURE, THERMAL AND FAST.
C    CXA1(L,K,IGROUP) - FAST ABS
C    CXA2(L,K,IGROUP) - THERMAL ABS
C    CXF1(L,K,IGROUP) - FAST NU-FISSION
C    CXF2(L,K,IGROUP) - THERMAL NU-FISSION
C    CXSC(L,K,IGROUP) - SLOWING
C    CXD1(L,K,IGROUP) - FAST DIFFUSION
C    CXD2(L,K,IGROUP) - THERMAL DIFFUSION
C
c Read loading and bundle data
```

Figure A.15. PWRCALC program listing.

```
      open(7,file='bundles.dat')
      OPEN(9,FILE='BUNDLES.PWR')
      write(9,*)
      read(7,*)nbun,ppmb,expi,expf,expinc
      do 10 i=1,nbun
      read(7,*,end=100)K,L,bid(L,K),idx(L,K),irot(L,K),
     $                 ((ex(2*L-3+M,2*K-3+N),N=1,2),M=1,2)
c Look for extent of map
      MAXL=max0(MAXL,L)
      MAXK=max0(MAXK,L)
      LNXB(L)=max0(LNXB(L),K)
      LNYB(K)=MAX0(LNYB(K),L)
c Categorize the number of unique fuel types in array index with nuniq entries
      read=.false.
      do 5 j=1,nuniq
 5    read=read.or.(index(j).eq.idx(L,K))
      if(.not.read) then
        index(nuniq)=idx(L,K)
        nuniq=nuniq+1
      endif
10    continue
      nuniq=nuniq-1
      close(7)
c Check for missing bundles
      do 20 L=1,MAXL
      do 20 K=1,LNXB(L)
20    if(idx(L,K).eq.0) goto 120
c Begin reading cross sections from library
      OPEN(8,FILE='BUNDLES.LIB',STATUS='OLD')
      nread=nuniq
 21   read(8,'(I2)',end=130)idxs
c Determine if idxs is member of array index
      read=.false.
      do 22 i=1,nuniq
 22   read=read.or.(index(i).eq.idxs)
      if(read) then
c Read cross section set
        read(8,*)(rcxa1(i),i=1,5)
        read(8,*)(rcxa2(i),i=1,5)
        read(8,*)(rcxf1(i),i=1,5)
        read(8,*)(rcxf2(i),i=1,5)
        read(8,*)(rcxsc(i),i=1,5)
        read(8,*)(rcxd1(i),i=1,5)
        read(8,*)(rcxd2(i),i=1,5)
c And assign to the bundles which require them
        do 25 L=1,MAXL
        do 25 K=1,lnxb(L)
        if(idx(L,K).eq.idxs) then
          do 27 i=1,5
          cxa1(L,K,i)=rcxa1(i)
          cxa2(L,K,i)=rcxa2(i)
          cxf1(L,K,i)=rcxf1(i)
          cxf2(L,K,i)=rcxf2(i)
          cxsc(L,K,i)=rcxsc(i)
          cxd1(L,K,i)=rcxd1(i)
27        cxd2(L,K,i)=rcxd2(i)
        endif
25      continue
c Then decrement the number of sets to read
        nread=nread-1
```

Figure A.15. PWRCALC program listing (continued).

```
      else
c Position forward to next set
        do 26 i=1,7
26      read(8,'(a1)')char
      endif
      if(nread.gt.0) goto 21
      close(8)
c All cross section retrieval completed
C Calculate nodal indicies and data (4 nodes per bundle and boundary water node)
      NYA=MAXL*2+1
      NXA=MAXK*2+1
c TFUEL is the total number of fueled nodes in the core (4*nbun - nodes beyond symmetry line)
      tfuel=nbun*4-2*(lnxb(1)+lnyb(1))+1
      DO 12 L=1,MAXL
      DO 12 M=1,2
      I=2*L-3+M
      LNX(I)=2*MAXO(LNXB(L)+1,LNXB(L-1))-1
12    LNXF(I)=LNXB(L)*2-1
      do 13 i=nya-1,nya
      LNX(I)=2*LNXB(MAXL)-1
13    LNXF(I)=0
      DO 14 K=1,MAXK
      DO 14 N=1,2
      J=2*K-3+N
14    LNY(J)=2*MAXO(LNYB(K)+1,LNYB(K-1))-1
      DO 15 J=NXA-1,NXA
15    LNY(J)=2*LNYB(MAXK)-1
C
      if(ppmb.lt.0) then
    search=.true.
    ppmb=0.0
      else
    search=.false.
      endif
C
16    CALL CSINIT(EX,RE)
      CALL RELPOW(RPOW)
C  Iterate on burnup
      if(expf.le.expi.or.expinc.le.0) goto 18
      expinc=amin1(expinc,expf-expi)
      do 17 I=0,nya-2
      do 17 J=0,lnxf(I)
17    ex(i,j)=ex(i,j)+expinc*rpow(i,j)
      expi=expi+expinc
      write(*,330)expi,expinc
      goto 16
c
c  Boron criticality search.  Stop on 1mk error
c
18    if(search.and.effk.ge.1.001) then
29      if(effk.ge.1.001) then
      ppmbo=ppmb
      ppmb=(effk-1)*1e4+ppmb
          effko=effk
    elseif(effk.le.0.999) then
      ppmb=(ppmbo-ppmb)*(1-effk)/(effko-effk)+ppmb
    elseif(effk.gt.0.999) then
      goto 19
        endif
    write(*,332) ppmb
```

Figure A.15. PWRCALC program listing (continued).

```
        CALL CSINIT(EX,RE)
        CALL RELPOW(RPOW)
     goto 29
       endif
C
C  Write the results to file BUNDLES.PWR
C
19     DO 30 L=1,MAXL
       DO 30 K=1,LNXB(L)
       K2M=2*(K-1)
       L2M=2*(L-1)
       WRITE(9,310) BID(L,K),K,L,IDX(L,K),IROT(L,K),
      $             ((ex(2*L-3+M,2*K-3+N),N=1,2),M=1,2),
      $             ((re(2*L-3+M,2*K-3+N),N=1,2),M=1,2),
      $             ((rpow(L2M+IL,K2M+IK),IK=0,1),IL=0,1)
       AVPWR=(RPOW(L2M  ,K2M  )+RPOW(L2M  ,K2M+1)
      $      +RPOW(L2M+1,K2M  )+RPOW(L2M+1,K2M+1))/4
       IF(AVPWR.GT.PWRMAX)THEN
           PWRMAX=AVPWR
           LMAX  =L
           KMAX  =K
       ENDIF
  30   continue
       WRITE(9,320) EFFK,PPMB,EXPI,KMAX,LMAX
       CLOSE(9)
       stop
c
c Errors
100    stop ' Unexpected end of file while reading problem data'
120    write(*,125)K,L
125    format(' Error: Missing bundle at column ',I1,' row ',I1)
       stop
130    stop ' Unexpected end of file while searching for cross sections'
310    FORMAT(1X,A,4(1X,I2),1X,12F9.5)
320    FORMAT(1X,F8.5,1X,F8.2,1X,F8.4,2I3)
330    FORMAT(/' ** Burn to ',F8.4,' GWD/MT in ',F8.5,' GWD/MT step **')
332    FORMAT(/' ** Boron criticality search: ',F8.2,' ppm **')
       END

       SUBROUTINE CSINIT(EX,RE)
C
C      ***  CSINIT CALCULATES BUNDLE CROSS SECTIONS ********
C
       COMMON/NUCLER/sigfnu(0:20,0:20,2),SIGS12(0:20,0:20)
      $,siga(0:20,0:20,2),D(0:20,0:20,2)
       COMMON/CXDATA/CXA1(8,8,5),CXA2(8,8,5),CXF1(8,8,5),CXF2(8,8,5),
      $              CXD1(8,8,5),CXD2(8,8,5),CXSC(8,8,5),PPMB
       COMMON/DIMEN/NXA,NYA,LNXF(0:20),LNX(0:20),LNY(0:20),AREA,TFUEL,
      $             CC(2),EXT(2),LNXB(0:8),maxl,maxk
       DIMENSION EX(0:20,0:20),RE(0:20,0:20),dwater(2),sigaw(2),aa(2)
       EQUIVALENCE (VOL,AREA)
c  Node size in cm.  Must be one quarter the bundle width.
       parameter(size=10.75)
c  Water and boron related cross sections
       parameter(xnhyd=.02938342,cxbmic=2087.445,h2oden=.711611,
      $ XNH2O=XNHYD/2.,sigslw=.3425255e-1)
       data dwater /1.911565,.2866381/,sigaw /.5686678e-3,.9693371e-2/
c  Maximum Borreson flux averaging
       data aa /2*0./
C
```

Figure A.15. PWRCALC program listing (continued).

```
      XNB=XNH2O*18./10.8*PPMB*.198E-6
      CXBMC=CXBMIC*XNB
      XNBR=.6022*H2ODEN/10.8*PPMB*.198E-6
      cxbmcr=CXBMIC*XNBR
      AREA=SIZE**2
      DO 15 K=1,2
   15 CC(K)=(1.-AA(K))/4.
C
C     *** CALCULATE CROSS SECTIONS FROM POLYNOMIAL FITS BY EXPOSURE **
C
      DO 130 L=1,MAXL
      DO 130 K=1,LNXB(L)
      DO 130 M=1,2
      DO 130 N=1,2
      I=2*L-3+M
      J=2*K-3+N
      EX1=EX(I,J)
      EX2=EX1*EX1
      EX3=EX2*EX1
      EX4=EX3*EX1
      SIGA(I,J,1)  =CXA1(L,K,1)    +CXA1(L,K,2)*EX1+CXA1(L,K,3)*EX2
     1             +CXA1(L,K,4)*EX3+CXA1(L,K,5)*EX4
      SIGA(I,J,2)  =CXA2(L,K,1)    +CXA2(L,K,2)*EX1+CXA2(L,K,3)*EX2
     1             +CXA2(L,K,4)*EX3+CXA2(L,K,5)*EX4+CXBMC
      SIGFNU(I,J,1)=CXF1(L,K,1)    +CXF1(L,K,2)*EX1+CXF1(L,K,3)*EX2
     1             +CXF1(L,K,4)*EX3+CXF1(L,K,5)*EX4
      SIGFNU(I,J,2)=CXF2(L,K,1)    +CXF2(L,K,2)*EX1+CXF2(L,K,3)*EX2
     1             +CXF2(L,K,4)*EX3+CXF2(L,K,5)*EX4
      SIGS12(I,J)  =CXSC(L,K,1)    +CXSC(L,K,2)*EX1+CXSC(L,K,3)*EX2
     1             +CXSC(L,K,4)*EX3+CXSC(L,K,5)*EX4
      D(I,J,2)     =CXD2(L,K,1)    +CXD2(L,K,2)*EX1+CXD2(L,K,3)*EX2
     1             +CXD2(L,K,4)*EX3+CXD2(L,K,5)*EX4
      D(I,J,1)     =CXD1(L,K,1)    +CXD1(L,K,2)*EX1+CXD1(L,K,3)*EX2
     1             +CXD1(L,K,4)*EX3+CXD1(L,K,5)*EX4
C
C     ***  TWO GROUP REACTIVITY CALCULATION WITH ZERO BUCKLING  ******
C
  120 RE(I,J)=(SIGFNU(I,J,2)/SIGA(I,J,2)*SIGS12(I,J)+SIGFNU(I,J,1))
     1  /(SIGS12(I,J)+ SIGA(I,J,1))
C
C     ***  NODAL VOLUME WHTED CALC OF PARAMETERS FOR RELPOW  *********
C
      SIGA(I,J,1)=(SIGA(I,J,1)+SIGS12(I,J))*VOL
      SIGA(I,J,2)=SIGA(I,J,2)*VOL
      SIGS12(I,J)=SIGS12(I,J)*VOL
      SIGFNU(I,J,1)=SIGFNU(I,J,1)*VOL
  130 SIGFNU(I,J,2)=SIGFNU(I,J,2)*VOL
C
C     ***  CROSS SECTIONS FOR WATER BOUNDARIES ***
C
      DO 133 I=1,NYA
      DO 133 J=LNXF(I)+1,LNX(I)
      D(I,J,1)=DWATER(1)
      D(I,J,2)=DWATER(2)
      SIGA(I,J,1)=(SIGAW(1)+SIGSLW)*VOL
      SIGA(I,J,2)=(SIGAW(2)+cxbmcr)*VOL
      SIGFNU(I,J,1)=0.
      SIGFNU(I,J,2)=0.
  133 SIGS12(I,J)=SIGSLW*VOL
```

Figure A.15. PWRCALC program listing (continued).

```
      DO 135 L=1,2
135   EXT(L)=0.71*(3.*DWATER(L))
C
      RETURN
      END

      SUBROUTINE RELPOW(RPOW)
C     ** RELPOW CALCULATES THE RELATIVE POWER FOR EACH NODE  *********
C     ** USING A 2 GROUP COURSE MESH DIFFUSION THEORY MODEL  *******
C
      COMMON/NUCLER/sigfnu(0:20,0:20,2),SIGS12(0:20,0:20)
     $,siga(0:20,0:20,2),D(0:20,0:20,2)
      COMMON/DIMEN/NXA,NYA,LNXF(0:20),LNX(0:20),LNY(0:20),AREA,TFUEL,
     $             CC(2),EXT(2),LNXB(0:8),maxl,maxk
      COMMON/ADI/EFFKS,TF(20,20,2),EFFK,RLAMS,TLFSLS,UK(20,20,2)
     $ ,LK(20,20,2),DK2(20,20,2),UJ(20,20,2),LJ(20,20,2),DJ1(20,20,2)
     $,DJ2(20,20,2),DK1(20,20,2)
      REAL LK,LJ
      EQUIVALENCE (SAVE,PHID)
      DIMENSION RPOW(0:20,0:20),B2(20,20),B4(20,20),PHID(20,20),
     $ SAVE(20,20),PHI(0:20,0:20,2),PHIA(20,20,2),PHIS(0:20,0:20,2),
     $ aa(2)
c
c  Problem specific constants
      parameter(maxo=40,testot=.000001,alphao=1.7)
c  Node size in cm.  Must be one quarter the bundle width
      parameter(size=10.75)
c  Maximum Borreson flux averaging
      data aa /2*0./
C
      TOTLFIS(I,J)=SIGFNU(I,J,1)*PHI(I,J,1)+SIGFNU(I,J,2)*PHI(I,J,2)
C
C SET FLUXES TO BE ZERO ON THE BOUNDARY
      DO 5 I=1,NXA+1
      DO 5 L=1,2
      DO 5 J=LNX(I),LNX(I)+1
      PHI(0,0,L)=0.
      PHI(0,I,L)=0.
      PHI(I,0,L)=0.
      PHI(J,I,L)=0.
      PHI(I,J,L)=0.
      PHIS(0,0,L)=0.
      PHIS(0,I,L)=0.
      PHIS(I,0,L)=0.
      PHIS(J,I,L)=0.
  5   PHIS(I,J,L)=0.
C
C     ****  CALCULATE CONSTANTS LK(I,J) THRU  ****
C     ****  UJ(I,J) FOR EACH POSITION IN CORE ****
C
      WF=0.
      DO 195 L=1,2
C
      LK(1,1,L)=0.
      UK(1,1,L)=2.*D(1,1,L)*D(1,2,L)/(D(1,1,L)+D(1,2,L))
      LJ(1,1,L)=0.
      UJ(1,1,L)=2.*D(1,1,L)*D(2,1,L)/(D(1,1,L)+D(2,1,L))
C
      DO 140 I=2,NYA
      LK(I,1,L)=0.
```

Figure A.15. PWRCALC program listing (continued).

```
      UK(I,1,L)=2.*D(I,1,L)*D(I,2,L)/(D(I,1,L)+D(I,2,L))
      LJ(I,1,L)=UJ(I-1,1,L)
140   UJ(I,1,L)=2.*D(I,1,L)*D(I+1,1,L)/(D(I,1,L)+D(I+1,1,L))
C
      DO 150 J=2,NXA
      LK(1,J,L)=UK(1,J-1,L)
      UK(1,J,L)=2.*D(1,J,L)*D(1,J+1,L)/(D(1,J,L)+D(1,J+1,L))
      LJ(1,J,L)=0.
150   UJ(1,J,L)=2.*D(1,J,L)*D(2,J,L)/(D(1,J,L)+D(2,J,L))
C
      DO 160 I=2,NYA
      DO 160 J=2,LNX(I)
      LK(I,J,L)=UK(I,J-1,L)
      UK(I,J,L)=2.*D(I,J,L)*D(I,J+1,L)/(D(I,J,L)+D(I,J+1,L))
      LJ(I,J,L)=UJ(I-1,J,L)
160   UJ(I,J,L)=2.*D(I,J,L)*D(I+1,J,L)/(D(I,J,L)+D(I+1,J,L))
C
      DO 170 I=1,NYA
      J=LNX(I)
170   UK(I,J,L)=D(I,J,L)*SIZE/(.5*SIZE+EXT(L))
C
      DO 180 J=1,NXA
      I=LNY(J)
180   UJ(I,J,L)=D(I,J,L)*SIZE/(.5*SIZE+EXT(L))
C
      DO 195 I=1,NYA
      DO 195 J=1,LNX(I)
      DJ2(I,J,L)=  LJ(I,J,L)+UJ(I,J,L)-WF
      DJ1(I,J,L)=-(LJ(I,J,L)+UJ(I,J,L)+SIGA(I,J,L)+WF)
      DK2(I,J,L)=-(LK(I,J,L)+UK(I,J,L)+SIGA(I,J,L)+WF)
195   DK1(I,J,L)=  LK(I,J,L)+UK(I,J,L)-WF
C
      EFFK=1.0
      EFFKS=1.0
      TLFISL=0.
      DO 100 I=1,NYA
      DO 100 J=1,LNX(I)
      PHI(I,J,1)=1.0
      PHI(I,J,2)=SIGS12(I,J)/SIGA(I,J,2)
      TF(I,J,1)=TOTLFIS(I,J)
100   TLFISL=TLFISL+TF(I,J,1)
      TLFSLS=TLFISL
C
      ITOUT=0
      WRITE(*,400)
C
C     **** FLUX SOLUTION ****
C
191   CALL FLXSLV(PHI,PHIS)
C
C     **** CALCULATION OF EFFECTIVE K ****
C
      TLFIS=0.
      DO 196 I=1,NYA
      DO 196 J=1,LNXF(I)
196   TLFIS=TLFIS+TOTLFIS(I,J)
      RLAM=TLFIS/TLFISL
      EFFK=EFFK*RLAM
      TEST0=ABS((TLFIS-TLFISL)/TLFIS)
      TLFISL=TLFIS
```

Figure A.15. PWRCALC program listing (continued).

```
      ITOUT=ITOUT+1
C
C     **** SOURCE ACCELERATION BY OVER RELAXATION ****
C
      IF(ITOUT.LE.5) THEN
       ALPHA=1.0
      ELSE
       ALPHA=ALPHAO
      ENDIF
      IF (ITOUT.NE.1) THEN
         TFISOR=0.
         DO 200 I=1,NYA
         DO 200 J=1,LNX(I)
         SAVE(I,J)=SAVE(I,J)+ALPHA*(TOTLFIS(I,J)-SAVE(I,J))
200      TFISOR=TFISOR+SAVE(I,J)
         TEMP=TLFIS/TFISOR/EFFK
         DO 240 I=1,NYA
         DO 240 J=1,LNX(I)
         TF(I,J,1)=SAVE(I,J)*TEMP
240      SAVE(I,J)=TOTLFIS(I,J)
      ELSE
         DO 220 I=1,NYA
         DO 220 J=1,LNX(I)
         SAVE(I,J)=TOTLFIS(I,J)
220      TF(I,J,1)=SAVE(I,J)/EFFK
      ENDIF
C
C     **** CHECK CONVERGENCE AND PRINT OUT DATA ****
C
      IF((ITOUT/3)*3+1.EQ.ITOUT)WRITE(*,410)ITOUT,EFFK,EFFKS,PHI(1,1,1),
     $                          PHI(4,4,1),TLFIS,TESTO,RLAM,RLAMS
250   IF(ITOUT.LE.MAXO.AND.TESTO.GT.TESTOT) GO TO 191
255   IF(ITOUT.GT.MAXO) WRITE(*,*) 'WARNING: MAXMIMUM OUTER ITERATIONS E
     $XCEEDED.  CALCULATION CONTINUES...'
C
C     **** AVERAGE FLUX OVER NODE USING INPUT AVERAGING FACTORS ****
C
      DO 340 K=1,2
      DO 270 I=1,NYA
      DO 270 J=1,LNX(I)
270   PHID(I,J)=PHI(I,J,K)*D(I,J,K)
C
      B2(1,1)=(PHID(1,1)+PHID(1,2))/(D(1,1,K)+D(1,2,K))
      B4(1,1)=(PHID(1,1)+PHID(2,1))/(D(1,1,K)+D(2,1,K))
      PHIA(1,1,K)=AA(K)*PHI(1,1,K)
     1           +CC(K)*(2.*PHI(1,1,K)+B2(1,1)+B4(1,1))
C
      DO 280 I=2,NYA
      B2(I,1)=(PHID(I,1)+PHID(I,2))/(D(I,1,K)+D(I,2,K))
      B4(I,1)=(PHID(I,1)+PHID(I+1,1))/(D(I,1,K)+D(I+1,1,K))
280   PHIA(I,1,K)=AA(K)*PHI(I,1,K)
     1           +CC(K)*(PHI(I,1,K)+B2(I,1)+B4(I-1,1)+B4(I,1))
C
      DO 290 J=2,NXA
      B2(1,J)=(PHID(1,J)+PHID(1,J+1))/(D(1,J,K)+D(1,J+1,K))
      B4(1,J)=(PHID(1,J)+PHID(2,J))/(D(1,J,K)+D(2,J,K))
290   PHIA(1,J,K)=AA(K)*PHI(1,J,K)
     1           +CC(K)*(PHI(1,J,K)+B2(1,J-1)+B2(1,J)+B4(1,J))
C
      DO 300 I=2,NYA
```

Figure A.15. PWRCALC program listing (continued).

```
      DO 300 J=2,LNX(I)
      B2(I,J)=(PHID(I,J)+PHID(I,J+1))/(D(I,J,K)+D(I,J+1,K))
      B4(I,J)=(PHID(I,J)+PHID(I+1,J))/(D(I,J,K)+D(I+1,J,K))
300   PHIA(I,J,K)=AA(K)*PHI(I,J,K)
     1           +CC(K)*(B2(I,J-1)+B2(I,J)+B4(I-1,J)+B4(I,J))
C
      DO 323 I=1,NYA
      JN=LNX(I)
323   B2(I,JN)=(PHI(I,JN,K)*EXT(K))/(EXT(K)+.5*SIZE)
      PHIA(1,NXA,K)=AA(K)*PHI(1,NXA,K)
     1             +CC(K)*(PHI(1,NXA,K)+B2(1,NXA-1)+B2(1,NXA)+B4(1,NXA))
C
      DO 324 I=2,NYA
      JN=LNX(I)
324   PHIA(I,JN,K)=AA(K)*PHI(I,JN,K)
     1            +CC(K)*(B2(I,JN-1)+B2(I,JN)+B4(I-1,JN)+B4(I,JN))
C
      DO 325 J=1,NXA
      IN=LNY(J)
325   B4(IN,J)=(PHI(IN,J,K)*EXT(K))/(EXT(K)+.5*SIZE)
      PHIA(NYA,1,K)=AA(K)*PHI(NYA,1,K)
     1             +CC(K)*(PHI(NYA,1,K)+B2(NYA,1)+B4(NYA-1,1)+B4(NYA,1))
C
      DO 326 J=2,NXA
      IN=LNY(J)
326   PHIA(IN,J,K)=AA(K)*PHI(IN,J,K)
     1            +CC(K)*(B2(IN,J-1)+B2(IN,J)+B4(IN-1,J)+B4(IN,J))
C
340   CONTINUE
C
C     **** CALCULATION OF RELATIVE POWER FOR EACH NODE ****
C
348   TLPOW=0.
      DO 350 I=1,NYA
      DO 350 J=1,LNX(I)
      RPOW(I,J)=PHIA(I,J,1)*SIGFNU(I,J,1)+PHIA(I,J,2)*SIGFNU(I,J,2)
350   TLPOW=TLPOW+RPOW(I,J)
      WRITE(*,420)
      AVGPOW=TLPOW/TFUEL
      WRITE(*,430)
      DO 370 I=1,NYA
      DO 360 J=1,LNXF(I)
360   RPOW(I,J)=RPOW(I,J)/AVGPOW
370   WRITE(*,440) I,(RPOW(I,J),J=1,LNXF(I))
C
C  Copy power values across symmetry lines
C
C  Center
      rpow(0,0)=rpow(1,1)
      rpow(1,0)=rpow(1,1)
      rpow(0,1)=rpow(1,1)
C  Vertical line of symmetry
      Do 380 I=2,nya
380   rpow(I,0)=rpow(I,1)
C  Horizontal line of symmetry
      Do 390 J=2,nxa
390   rpow(0,J)=rpow(1,J)
      RETURN
C
400   FORMAT (/' ITOUT  K-EFF  K-EFF* PHI1(1,1) PHI1(4,4) FISSIONS CONV
```

Figure A.15. PWRCALC program listing (continued).

```
     1ERGENCE  LAMBDA  LAMBDA*')
410   FORMAT (' ',I5,2F8.5,2F9.5,2E11.4,2F9.6)
420   FORMAT (//,10X,' *****  RELATIVE POWER FOR I,J POSITION  *****'/)
430   FORMAT(6X,'1    2    3    4    5    6    7    8    9   10   11   1
     $2   13   14   15',/)
440   FORMAT(' ',I2,1X,17F5.2)
      END

      SUBROUTINE FLXSLV(PHI,PHIS)
C  SOLVES THE ADI EQUATIONS USING GAUSSIAN ELIMINATION FOR A FULL TIME STEP
C    UK,DK,LK,UJ,DJ,LJ - COEFFICIENTS FOR ADI EQUATIONS (IN)
C    PHI               - FULL TIME STEP FLUX ARRAY (OUT,IN)
C    PHIS              - HALF TIME STEP FLUX ARRAY (OUT,IN)
      COMMON/ADI/EFFKS,TF(20,20,2),EFFK,RLAMS,TLFSLS,UK(20,20,2)
     $ ,LK(20,20,2),DK2(20,20,2),UJ(20,20,2),LJ(20,20,2),DJ1(20,20,2)
     $,DJ2(20,20,2),DK1(20,20,2)
      REAL LK,LJ
      COMMON/DIMEN/NXA,NYA,LNXF(0:20),LNX(0:20),LNY(0:20),AREA,TFUEL,
     $            CC(2),EXT(2),LNXB(0:8),maxl,maxk
      COMMON/NUCLER/sigfnu(0:20,0:20,2),SIGS12(0:20,0:20)
     $,siga(0:20,0:20,2),D(0:20,0:20,2)
      DIMENSION PHI(0:20,0:20,2),B(20),PHIS(0:20,0:20,2)
      EXTERNAL SRCE1,SRCE2
C
      RHS1(K,J,L)=-UK(K,J,L)*PHI (K,J+1,L)+DK1(K,J,L)*PHI (K,J,L)
     $           -LK(K,J,L)*PHI (K,J-1,L)-SRCE1(PHI,PHIS,K,J,L)
      RHS2(K,J,L)=-UJ(K,J,L)*PHIS(K+1,J,L)+DJ2(K,J,L)*PHIS(K,J,L)
     $           -LJ(K,J,L)*PHIS(K-1,J,L)-SRCE2(PHI,PHIS,K,J,L)
C
C  COMPUTE FLUXES AT END OF HALF TIME STEP IMPLICIT BY ROWS
      DO 70 L=1,2
      DO 70 J=1,NXA
      B(1)=DJ1(1,J,L)
      PHIS(1,J,L)=RHS1(1,J,L)/B(1)
C  FORWARD GUASSIAN ELIMINATION
      DO 65 K=2,LNY(J)
      B(K)=DJ1(K,J,L)-LJ(K,J,L)*UJ(K-1,J,L)/B(K-1)
   65 PHIS(K,J,L)=(RHS1(K,J,L)-LJ(K,J,L)*PHIS(K-1,J,L))/B(K)
C BACK SOLUTION
      DO 70 K=LNY(J)-1,1,-1
   70 PHIS(K,J,L)=PHIS(K,J,L)-UJ(K,J,L)*PHIS(K+1,J,L)/B(K)
C
C COMPUTE INTERMEDIATE MULTIPLICATION FACTOR
C
      TLFISS=0.
      DO 90 I=1,NYA
      DO 90 J=1,LNXF(I)
   90 TLFISS=TLFISS+SIGFNU(I,J,1)*PHIS(I,J,1)+SIGFNU(I,J,2)*PHIS(I,J,2)
      RLAMS=TLFISS/TLFSLS
      EFFKS=EFFKS*RLAMS
      TLFSLS=TLFISS
C
C COMPUTE FLUXES AT END OF FULL TIME STEP IMPLICIT BY COLUMNS
C
      DO 130 L=1,2
      DO 130 K=1,NYA
      B(1)=DK2(K,1,L)
      PHI(K,1,L)=RHS2(K,1,L)/B(1)
C  FORWARD GUASSIAN ELIMINATION
      DO 110 J=2,LNX(K)
```

Figure A.15. PWRCALC program listing (continued).

```
      B(J)=DK2(K,J,L)-LK(K,J,L)*UK(K,J-1,L)/B(J-1)
  110 PHI(K,J,L)=(RHS2(K,J,L)-LK(K,J,L)*PHI(K,J-1,L))/B(J)
C BACK SOLUTION
      DO 130 J=LNX(K)-1,1,-1
  130 PHI(K,J,L)=PHI(K,J,L)-UK(K,J,L)*PHI(K,J+1,L)/B(J)
      RETURN
      END

      FUNCTION SRCE1(PHI,PHIS,K,J,L)
C  RETURNS THE SOURCE VALUE OF THE RIGHT HAND SIDE OF THE ADI EQUATIONS
C    PHI    - FULL INTERVAL FLUX ARRAY (IN)
C    PHIS   - HALF INTERVAL FLUX ARRAY (IN)
C    K,J    - COLUMN, ROW INDICES (IN)
C    L      - GROUP NUMBER
      COMMON/NUCLER/sigfnu(0:20,0:20,2),SIGS12(0:20,0:20)
     $,siga(0:20,0:20,2),D(0:20,0:20,2)
      COMMON/ADI/EFFKS,TF(20,20,2),EFFK,RLAMS,TLFSLS,UK(20,20,2)
     $ ,LK(20,20,2),DK2(20,20,2),UJ(20,20,2),LJ(20,20,2),DJ1(20,20,2)
     $,DJ2(20,20,2),DK1(20,20,2)
      REAL LK,LJ
      DIMENSION PHI(0:20,0:20,2),PHIS(0:20,0:20,2)
C
      IF(L.EQ.1) SRCE1=TF(K,J,L)
      IF(L.EQ.2) SRCE1=SIGS12(K,J)*PHIS(K,J,1)
      RETURN
C
      ENTRY SRCE2(PHI,PHIS,K,J,L)
      IF(L.EQ.1) SRCE2=(SIGFNU(K,J,1)*PHIS(K,J,1)
     $                 +SIGFNU(K,J,2)*PHIS(K,J,2))/EFFKS
      IF(L.EQ.2) SRCE2=SIGS12(K,J)*PHI(K,J,1)
      RETURN
      END
```

Figure A.15. PWRCALC program listing (continued).

presents the source listing for PWRCALC.

Comparisons of PWRCALC were made with the previous nodal calculation which used successive displacement for the inner iteration. A substantial improvement in consistency of the power distribution was obtained by the ADI inner iteration in PWRCALC within a running time identical to the prior nodal code. Before powers at nodes about the map could vary as much as 20% from repeated calculations with identical loadings however with PWRCALC the results remained consistent.

REFERENCES

Anderson, Dale A., John C. Tannehill, and Richard H. Pletcher. Computational Fluid Mechanics and Heat Transfer, Hemisphere Publishing Corp., 1984, 136.

Barry, R. F. "LEOPARD--a spectrum-dependent non-spacial depletion code for the IBM-7094," USAEC Report WCAP-6058, 1964.

Carnahan, Brice, H. A. Luther, and James O. Wilkes, Applied Numerical Methods, John Wiley and Sons, 1969, 508.

Chao, Y. A., and J. A. Penkrot. "Diffusive homogeneity--the principle of the superfast multidimensional nodal code, SUPERNOVA," ANS Transactions, 55 (1987), 583-4.

Gladwell, I and R. Wait, ed. A Survey of Numerical Methods for Partial Differential Equations, Clarendon Press, 1979, 276-294.

Wachspress, Eugene L. Iterative Solution of Elliptic Systems and Applications to the Neutron Diffusion Equations of Reactor Physics, Prentice-Hall, 1966.

APPENDIX B

LIST OF CONSTRAINTS

The following pages of this appendix list the constraints documented earlier in chapter IV which are imposed to solve the core reload problem. The constraints include the additional restriction formulated in chapter VII on exchanges with bundles of high reactivity difference. The listings belong to two classes: constraints imposed upon moving bundles in the loading and the solution objective function which is itself a constraint. The objective function specifies the goal the solution is to achieve and following each solution step it is used to evaluate for the better of two loadings. The objective function is found in class Core. The remaining constraints restrict moves which change bundle map position or rotation for either single bundles or two bundles involved in an exchange. Most these constraints are common to each strategy and belong to class Constraints, however specific constraints may be found in each strategy (e.g. old fuel strategy). Additional constraints which differentiate bundle location in a pool or loading may also be imposed in classes Loading, Bundle, and Pool.

```
(Search) subclass:              #Constraints
instanceVariableNames:
classVariableNames:
poolDictionaries:


Constraints class methods


Constraints methods


moveValid:aBundle
"Answer true for bundles which satisfy position constraints"

"Do not move the center bundle"
aBundle position = (10@1)
    ifTrue:[^false]
    explainTrue:['Do not exchange the center bundle'].

"Any peripheral bundles are not moved"
(aBundle position region = Periphery)
    ifTrue:[^false]
    explainTrue:['Do not move peripheral bundle ',aBundle name].

^true


rotateValid:aBundle
"Perform some tests for constraints on core power symmetry and bundle
reactivity assymmetry"
"Answer false if the bundle is on an axis of symmetry"
(aBundle position x = 1) | (aBundle position y = 1)
    ifTrue:[^false]
    explainTrue:['Do not rotate ',aBundle name,' on an axis of symmetry'].

"Answer true if the ratio of maximum to minimum corner
reactivity is greater than a critical value."
aBundle reactivity gradient > 1.001
    ifFalse:[^false]
    explainFalse:['Do not rotate ',aBundle name,
                ' because of\     insignificant reactivity gradient'].
^true
```

```
xchange:bun1 validTo:bun2
"Answer true for bundle exchanges with proper parity"
|region1 region2|
"Don't attempt to exchange the bundle with itself"
bun1 = bun2                              ifTrue:[^false].

"Satisy single position constraints on each bundle"
(self moveValid:bun1) &
(self moveValid:bun2)
    ifFalse:[^false].

region1:=bun1 position region.
region2:=bun2 position region.

"Exchange even parity bundles with odd parity bundles"
(region1 = Odd) & (region2 = Even) ifTrue:[^true].
(region1 = Even) & (region2 = Odd) ifTrue:[^true].

"Don't exchange like parity bundles unless they are
intermediate parity"
(region1 = region2) &
    (region1 ~= Intermediate)
    ifTrue:[^false]
    explainTrue:['Do not exchange like parity bundles ',bun1 name,
                ' and ',bun2 name].

"Exchange intermediate bundles with all bundle types but
peripheral"
(region1 = Intermediate) &
    (region2 ~= Periphery)          ifTrue:[^true].
(region1 ~= Periphery) &
    (region2 = Intermediate)        ifTrue:[^true].

self error:'xchange:validTo: no condition match'
```

```
(Constraints) subclass:       #OldFuel
instanceVariableNames:
classVariableNames:
poolDictionaries:


OldFuel class methods


OldFuel methods

xchange:bun1 validTo:bun2
"Reject bundles of power greater than bun1."
 (bun2 power average) > (bun1 power average)
    ifTrue:[^false]
    explainTrue:['Do not exchange ',bun1 name,' since its power exceeds ',
                bun2 name].

"Reject moves with bundles whose reactivity difference
 is greater than 0.18"
((bun1 reactivity average) - (bun2 reactivity average)) abs > 0.18
    ifTrue:[^false]
    explainTrue:['Do not exchange ',bun1 name,' and ',bun2 name,
                ' because their\     reactivity difference exceeds 0.18'].

"Accept exchanges of bun1 with lower power and reactivity bundles only"
((bun1 power average) > (bun2 power average)) &
((bun1 reactivity average) < (bun2 reactivity average))
    ifTrue:[^false]
    explainTrue:['Do not exchange ',bun1 name,' since its reactivity\     ',
                'is less than low power bundle ',bun2 name]
    ifFalse:[^super xchange:bun1 validTo:bun2].

(Object) subclass:            #Core
instanceVariableNames:
classVariableNames:
poolDictionaries:


Core class methods


objectiveFunction
"Answer a block for the objective of the solution which is used to determine
the better of two loadings"
"Minimum power peaking constraint -- answer true if the local power peak of
loading is less than the local power peak of best"
^[:loading :best|loading maxBundle power average < best maxBundle power average]
```

APPENDIX C

LIST OF RULES

This appendix lists the rules documented in chapter IV which generate moves and direct the solution of the core reload problem. The rules are associated with the new fuel, old fuel, and rotate fuel strategies. Also included in the listing are the rules of class loading which generate moves about either the one quarter or one eighth core symmetry lines such that bundle placement is symmetrical and power distribution is balanced. These last rules were written to "clone" a bundle if an exchange about a line of symmetry requires matching bundles to be placed in mirror positions. Cloned bundles are identical in all attributes but position to their predecessors and have a 'c' appended to their name.

Smalltalk methods associated with move generation for the new fuel strategy are included in the listing as procedures not rules since they include heuristics coded from operations which cannot be stated in an If...Then format. The two methods--minBundlesAbout: and movesFor:withMin:--are needed to select and order potential exchanges with bundles surrounding the maximum power bundle.

```
(Constraints) subclass:        #NewFuel
instanceVariableNames:         loading
                               localBundles
                               considerFirst
                               bundlesByPower
                               minBundles
classVariableNames:
poolDictionaries:


NewFuel class methods


NewFuel methods


evaluate:aLoading
"Evaluates if the receiver process has a loading whose maximum
power bundle is less than the best loading.  Answer the loading in
any case."
|point1 point2|
(evalBlock value:aLoading value:manager bestLoading)
    ifTrue: [^manager nextLoading:aLoading;
                      bestLoading:aLoading]
"If the new loading has a bundle of maximum power in excess of the manager bestLoading
and the maximum power bundles are in transpose positions, then continue the
search with the new loading for one step."
    ifFalse:[point1:=aLoading maxBundle position.
             point2:=manager bestLoading maxBundle position.
             (point1 transpose = point2) & (point1 ~= point2)
                 ifTrue:[manager nextLoading:aLoading.
                         ^aLoading].
             manager nextLoading:(manager bestLoading).
            ^aLoading].
```

```
generateMovesFor:aLoading
"Use heuristics to generate additional moves for the aLoading movesToApply list"
(loading isNil)|(aLoading ~= loading)
    ifTrue:[loading:=aLoading.
            minBundles:=self minBundlesAbout:aLoading maxBundle.
            bundlesByPower:=aLoading contents
                    asSortedCollection:[:a :b|(a power average)
                                        >= (b power average)].
            ruleToApply:=1].
ruleToApply > 5
    ifFalse:[ruleToApply = 2
                    ifTrue: [self movesFor:(bundlesByPower at:2)
                                withMin:((self minBundlesAbout:
                                        (bundlesByPower at:2)) first)]
                    ifFalse:[self movesFor:aLoading maxBundle
                                withMin:minBundles removeFirst].
                ruleToApply:=ruleToApply+1.
                ^true].
^ready:=false


minBundlesAbout:aBundle
"Answer a SortedCollection of bundles ordered by increasing power
in the neighborhood of extent 2 from aBundle"
^(aBundle neighborhoodOf:(loading contents) extent:(2@2))
    asSortedCollection:[:a :b|(a power average)
                        <= (b power average)].


movesFor:maxBundle withMin:minBundle
"Generate all moves of the maximum power bundle maxBundle with those bundles
immediately surrounding it and nearest to the local minimum minBundle.
If no bundles to be exchanged satisfy constraints report failure."
|bundlesNearMaxToExchange bundlesNearestMin|
bundlesNearMaxToExchange:=((maxBundle neighborhoodOf:(loading contents) extent:(1@1))
                    select:[:bundle|self xchange:maxBundle
                                    validTo:bundle    ])
                    asSortedCollection:[:a :b|(minBundle distanceTo:a)
                                        <= (minBundle distanceTo:b)].
bundlesNearestMin:=bundlesNearMaxToExchange
                    select:[:bundle|(bundlesNearMaxToExchange first distanceTo:minBundle)
                                            =  (bundle distanceTo:minBundle)].
"Check reactivity of move and maximum bundle"
bundlesNearestMin
    do:[:bundle|bundle reactivity average > maxBundle reactivity average
                ifTrue:[^self movesFor:bundle
                                withMin:((self minBundlesAbout:bundle) first)]
"Add only moves unique to this loading"
                ifFalse:[loading
                        addUniqueMove:(loading exchange:maxBundle byOneEighthSymmetry:bundle)]].
```

```
(Constraints) subclass:       #OldFuel
instanceVariableNames:
classVariableNames:
poolDictionaries:


OldFuel class methods


OldFuel methods


generateMovesFor:aLoading
"Answer the moves of Shuffle B"
|localMaxBundle minBundles localBundles newMove|
ruleToApply = 1
    ifTrue:[localBundles:=(aLoading maxBundle neighborhoodOf:(aLoading contents)
                                        extent:(1@1))
                        select:[:bundle|self moveValid:bundle]].
ruleToApply = 2
    ifTrue:[localBundles:=(aLoading maxBundle neighborhoodOf:(aLoading contents)
                                        extent:(2@2))
                        select:[:bundle|self moveValid:bundle]].
ruleToApply > 2
    ifTrue:[^ready:=false].
localMaxBundle:=(localBundles
                    asSortedCollection:[:a :b|(a power average)
                                        >= (b power average)])
                    first.
minBundles:=(aLoading contents select:[:bundle|self xchange:localMaxBundle
                                                validTo:bundle])
            asSortedCollection:[:a :b|(a power average)
                                <= (b power average)].
minBundles do:[:bundle|aLoading addUniqueMove:(aLoading exchange:localMaxBundle
                                                    byOneQuarterSymmetry:bundle)].
ruleToApply:=ruleToApply+1.
^true
```

```
(Constraints) subclass:       #RotateFuel
instanceVariableNames:
classVariableNames:
poolDictionaries:


RotateFuel class methods


RotateFuel methods


evaluate:aLoading
"Evaluates if the receiver process has a loading whose maximum
power bundle is less than the best loading.  Answer the loading in
any case."
|nextLoading nextMoves|
(evalBlock value:aLoading value:manager bestLoading)
    ifTrue: [ "Keep remaining base of parent loading"
            nextLoading:=aLoading base:self;yourself.
             "If this new best loading was generated from a rotation move
              (a rotation move always has but one bundle in its addBundles or
               deleteBundles list) the power peak has not likely changed position.
              Copy any unused rotations from the parent loading's movesToApply list."
              aLoading moveTo addBundles size = 1
                   ifTrue:[nextMoves:=
                       (nextLoading moveTo parentLoading movesToApply
                            select:[:move|move addBundles size = 1])
                            collect:[:move|move copy newLocation:nextLoading].
                           nextLoading movesToApply:nextMoves].
            ^manager nextLoading:nextLoading;
                     bestLoading:nextLoading]
   "Otherwise continue with best aLoading but return new loading to add to
    load list."
    ifFalse:[manager nextLoading:(manager bestLoading).
           ^aLoading].
```

```
generateMovesFor:aLoading
"Generate the moves for Shuffle C"
|localBundles|
ruleToApply > 2
    ifTrue:[^ready:=false].
localBundles:=((aLoading maxBundle neighborhoodOf:(aLoading contents)
                                        extent:(1@1))
              select:[:bundle|self rotateValid:bundle]) asOrderedCollection.
"Add the maxBundle itself for the first try"
ruleToApply = 1
    ifTrue:[localBundles addFirst:(aLoading maxBundle)].
ruleToApply = 2
    ifTrue:[localBundles:=((aLoading maxBundle neighborhoodOf:(aLoading contents)
                                                    extent:(2@2))
                          reject:[:bundle|localBundles includes:bundle])
                          select:[:bundle|self rotateValid:bundle]].
"Add moves to rotate each bundle 180 degrees"
localBundles do:[:bundle|aLoading addUniqueMove:(aLoading rotate:bundle by:2)].
ruleToApply:=ruleToApply+1.
^true
```

```
(Core) subclass:              #Loading
instanceVariableNames:        name
                              contents
                              keff
                              shim
                              exposure
                              moveTo
                              maxBundle
                              movesApplied
                              movesToApply
                              base
classVariableNames:
poolDictionaries:
```

Loading class methods

Loading methods

```
exchange:bundle1 with:bundle2
"Answer a move for receiver loading representing the exchange
of bundle1 with bundle2."
^Move new
     addBundles:(Array with:(bundle1 copy position:bundle2 position)
                       with:(bundle2 copy position:bundle1 position));
     deleteBundles:(Array with: bundle1
                           with: bundle2);
     parentLoading:self;
     name:bundle1 name,'<>',bundle2 name
```

```
exchange:bundle1 byOneQuarterSymmetry:bundle2
"Answer a move for receiver loading representing the shuffle
of bundle1 with bundle2."
"If neither bundle1 and bundle2 are along the axis of symmetry or are
in transpose positions then swap both"
(bundle1 position >= (2@2) & (bundle2 position >= (2@2))
 or:[bundle1 position transpose = bundle2 position])
    ifTrue:[^Move new
                  addBundles:(Array with:(bundle1 copy position:bundle2 position)
                                    with:(bundle2 copy position:bundle1 position));
               deleteBundles:(Array with: bundle1
                                    with: bundle2);
               parentLoading:self;
                        name:bundle1 name,'<4>',bundle2 name].
"If both bundle1 and bundle2 are along the axis of symmetry
then swap bundle1 and bundle2 and their mirror bundles."
((bundle1 position x=1) | (bundle1 position y=1)) &
((bundle2 position x=1) | (bundle2 position y=1))
    ifTrue:[^Move new
                  addBundles:(Array with:(bundle1 copy  position:bundle2 position)
                                    with:(bundle2 copy  position:bundle1 position)
                                    with:(bundle1 clone position:bundle2 position transpose)
                                    with:(bundle2 clone position:bundle1 position transpose));
               deleteBundles:(Array with: bundle1
                                    with: bundle2
                                    with:(contents at:bundle1 position transpose)
                                    with:(contents at:bundle2 position transpose));
               parentLoading:self;
                        name:bundle1 name,'<4>',bundle2 name].
"If bundle1 is on an axis of symmetry and bundle2 is not then
perform a 'half swap' and duplicate bundle2 on the axis."
((bundle1 position x=1) | (bundle1 position y=1)) &
(bundle2 position >= (2@2))
     ifTrue:[^Move new
                  addBundles:(Array with:(bundle1 copy  position:bundle2 position)
                                    with:(bundle2 copy  position:bundle1 position)
                                    with:(bundle2 clone position:bundle1 position transpose));
               deleteBundles:(Array with: bundle1
                                    with: bundle2
                                    with:(contents at:bundle1 position transpose));
               parentLoading:self;
                        name:bundle1 name,'<4>',bundle2 name].
```

```
"If bundle2 is on an axis of symmetry and bundle1 is not then
perform a 'half swap' and duplicate bundle1 on the axis."
((bundle2 position x=1) | (bundle2 position y=1)) &
(bundle1 position >= (2@2))
     ifTrue:[^Move new
                  addBundles:(Array with:(bundle2 copy  position:bundle1 position)
                                  with:(bundle1 copy  position:bundle2 position)
                                  with:(bundle1 clone position:bundle2 position transpose));
                deleteBundles:(Array with: bundle2
                                  with: bundle1
                                  with:(contents at:bundle2 position transpose));
                parentLoading:self;
                        name:bundle2 name,'<4>',bundle1 name].

exchange:bundle1 byOneEighthSymmetry:bundle2
"Answer a move for receiver loading representing the shuffle
of bundle1 with bundle2."
|bundleA bundleB|
"If bundles are in transpose positions or on the 1/8th diagonal then swap both"
((bundle1 position transpose = bundle2 position)
 or:[(bundle1 position x = bundle1 position y) &
     (bundle2 position x = bundle2 position y)])
    ifTrue:[^Move new
                  addBundles:(Array with:(bundle1 copy position:bundle2 position)
                                  with:(bundle2 copy position:bundle1 position));
                deleteBundles:(Array with: bundle1
                                  with: bundle2);
                parentLoading:self;
                        name:bundle1 name,'<8>',bundle2 name].

"If bundle1 is on the 1/4 axis of symmetry and bundle2 is on the 1/8 axis of symmetry then
perform a 'half swap' and mirror bundle2 on the 1/4 axis."
((bundle1 position x=1) | (bundle1 position y=1)) &
 (bundle2 position x  = bundle2 position y)
     ifTrue:[^Move new
                  addBundles:(Array with:(bundle1 copy  position:bundle2 position)
                                  with:(bundle2 copy  position:bundle1 position)
                                  with:(bundle2 clone position:bundle1 position transpose));
                deleteBundles:(Array with: bundle1
                                  with: bundle2
                                  with:(contents at:bundle1 position transpose));
                parentLoading:self;
                        name:bundle1 name,'<8>',bundle2 name].
```

```
"If bundle2 is on the 1/4 axis of symmetry and bundle1 is on the 1/8 axis of symmetry then
perform a 'half swap' and mirror bundle1 on the 1/4 axis."
((bundle2 position x=1) | (bundle2 position y=1)) &
 (bundle1 position x  = bundle1 position y)
    ifTrue:[^Move new
                  addBundles:(Array with:(bundle2 copy  position:bundle1 position)
                                    with:(bundle1 copy  position:bundle2 position)
                                    with:(bundle1 clone position:bundle2 position transpose));
               deleteBundles:(Array with: bundle2
                                    with: bundle1
                                    with:(contents at:bundle2 position transpose));
               parentLoading:self;
                       name:bundle2 name,'<8>',bundle1 name].
"If bundle1 is on the one eighth axis of symmetry exchange bundle1 and bundle2
and place a clone of bundle1 in the transpose position of bundle2"
(bundle1 position x  = bundle1 position y) &
(bundle2 position x ~= bundle2 position y) &
(bundle2 position x ~= 1) & (bundle2 position y ~= 1)
    ifTrue:[^Move new
                  addBundles:(Array with:(bundle1 copy  position:bundle2 position)
                                    with:(bundle2 copy  position:bundle1 position)
                                    with:(bundle1 clone position:bundle2 position transpose));
               deleteBundles:(Array with: bundle1
                                    with: bundle2
                                    with:(contents at:bundle2 position transpose));
               parentLoading:self;
                       name:bundle1 name,'<8>',bundle2 name].
"If bundle2 is on the one eighth axis of symmetry exchange bundle2 and bundle1
and place a clone of bundle2 in the transpose position of bundle1"
(bundle2 position x  = bundle2 position y) &
(bundle1 position x ~= bundle1 position y) &
(bundle1 position x ~= 1) & (bundle1 position y ~= 1)
    ifTrue:[^Move new
                  addBundles:(Array with:(bundle2 copy  position:bundle1 position)
                                    with:(bundle1 copy  position:bundle2 position)
                                    with:(bundle2 clone position:bundle1 position transpose));
               deleteBundles:(Array with: bundle2
                                    with: bundle1
                                    with:(contents at:bundle1 position transpose));
               parentLoading:self;
                       name:bundle2 name,'<8>',bundle2 name]
```

```
"If either bundle1 or bundle2 are not on an axis of symmetry
then swap bundle1 and bundle2 and their mirror bundles."
   ifFalse:[bundleA:=contents at:bundle1 position transpose.
            bundleB:=contents at:bundle2 position transpose.
            ^Move new
                    addBundles:(Array with:(bundle1 copy position:bundle2 position)
                                        with:(bundle2 copy position:bundle1 position)
                                        with:(bundleA copy position:bundleB position)
                                        with:(bundleB copy position:bundleA position));
                 deleteBundles:(Array with: bundle1
                                        with: bundle2
                                        with: bundleA
                                        with: bundleB);
                 parentLoading:self;
                         name:bundle1 name,'<8>',bundle2 name].
```