Stanford Artificial Intelligence Project
MEMO AI-58

AN ADAPTIVE COMMAND AND CONTROL SYSTEM UTILIZING HEURISTIC LEARNING PROCESSES

BY

MONTI D. CALLERO

DECEMBER 1967

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

Stanford University Libraries
Dept. of Special Collections
Coll SC340 Title
Box 22 Series 1986-052
Fol 14 Fol. Title

AN ADAPTIVE COMMAND AND CONTROL SYSTEM UTILIZING HEURISTIC LEARNING PROCESSES

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF OPERATIONS RESEARCH

AND THE COMMITTEE ON THE GRADUATE DIVISION

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

By

Monti Don Callero

December 1967

STANFORD ARTIFICIAL INTELLIGENCE PROJECT
MEMO AI-58

December 1967

AN ADAPTIVE COMMAND AND CONTROL SYSTEM UTILIZING HEURISTIC LEARNING PROCESSES

by Monti D. Callero*

ABSTRACT: The objectives of the research reported here are to develop an automated decision process for real time allocation of defense missiles to attacking ballistic missiles in general war and to demonstrate the effectiveness of applying heuristic learning to seek optimality in the process. The approach is to model and simulate a missile defense environment and generate a decision procedure featuring a self-modifying, heuristic decision function which improves its performance with experience. The goal of the decision process that chooses between the feasible allocations is to minimize the total effect of the attack, measured in cumulative loss of target value. The goal is pursued indirectly by considering the more general problem of maintaining a strong defense posture, the ability of the defense system to protect the targets from both current and future loss.

Using simulation and analysis, a set of calculable features are determined which effectively reflect the marginal deterioration of defense posture for each allocation in a time interval. A decision function, a linear polynomial of the features, is evaluated for each feasible allocation and the allocation having the smallest value is selected. A heuristic learning process is incorporated in the model to evaluate the performance of the decision process and adjust the decision function coefficients to encourage correct comparison of alternative allocations. Simulated attacks presenting typical defense situations were cycled against the decision procedure with the result that the decision function coefficients converged under the learning process and the decision process became increasingly effective.

The research reported here was supported in part by the Advanced Research Projects Agency of the Office of the Secretary of Defense (SD-183).

*Now at Department of Astronautics and Computer Science, USAF Academy Colorado.

ACKNOWLEDGMENTS

Heartfelt appreciation is given to Professor Gerald J. Lieberman who, as principle advisor, has guided and encouraged the author through four years of graduate study, and Professor Edward A. Feigenbaum who undertook the direction of the research reported here.

Valuable suggestions and criticism were contributed by Professors Frederick S. Hillier and Peter R. Winters, who also read the final draft.

The final report was typed by Grace Mickelson.

Graduate study has been made possible by the United States Air Force under the Civilian Institutes Division of the Institute of Technology. This research was supported in part by the Advanced Research Projects Agency of the Office of the Secretary of Defense (SD-183).

INDEX

LIST OF TABLES

LIST OF ILLUSTRATIONS

I. INTRODUCTION

1.1 Objectives and Approach

This paper reports an artificial intelligence approach to the military command and control problem of real time allocation of defense missiles to attacking ballistic missiles in general war.

The defense missile allocation problem is representative of a large class of so called "ill-structured" management problems whose complexity places them beyond the scope of the powerful operations research algorithms, which are iterative by nature and easily adapted to digital computers. The impetus for studying the particular problem is that the real time aspect and extensive sphere of relevant data in an attack-defense environment makes it unreasonable to expect a human decision maker to remain abreast of the developing situation, which suggests that an automated decision procedure must be an integral part of a defense missile system if allocation is to be made in a systematic, optimum seeking way. The objectives of this research are to develop such a decision procedure for the allocation of defense missiles and demonstrate the effectiveness of applying heuristic learning to the solution of ill structured operations research problems. The approach is to simulate a missile defense environment and generate a decision procedure featuring a self-modifying, heuristic decision function which improves its performance with experience.

1.2 Background: Artificial Intelligence[1]

The development of general-purpose digital computers has brought with it a significant research effort toward the mechanization of decision making processes. Computer programs have been implemented which prove theorems, select stock portfolios, recognize patterns, play board games and perform other tasks which, if accomplished by humans, would be considered as requiring the use of intelligence. The accomplishment of intelligent activity by computers is what is generally known as Artificial Intelligence, and the goal of artificial intelligence research is "to construct computer programs which exhibit behavior that we call "intelligent behavior" where we observe it in human beings".[2] Such programs cannot respond in a stereotyped manner to all situations but must condition their performance to the stimuli of the task environment, often to the extent of recognizing the quality of their own responses and initiating internal adaptation to enhance future effectiveness.

A major effort in artificial intelligence research has been toward the discovery and application of heuristics. The word heuristic has various interpretations but most commonly refers to a rule of thumb, method or trick which improves the efficiency of a problem solving system, often at the expense of a guaranteed best answer. Examples of

---

[1] Only those elements of Artificial Intelligence which are directly applicable to this research are discussed. For a more complete, but concise, look at the subject see [1], especially the introduction to Part I and Minsky's Steps Toward Artificial Intelligence, pp. 406-450, in Part 3. This volume is a collection of research papers describing the more interesting developments in artificial intelligence up to 1962.

[2] Feigenbaum [1] p. 3.

heuristics abound. "Buy from the lowest bidder" and "always vote a straight ticket" are two which indisputably ease decision making, but admittedly do not always lead to the best results. When playing games people develop heuristics which seem to work pretty well much of the time, such as "always check, it may be mate", "never open with less than thirteen points" and "protect the queen at any cost", and eliminate large numbers of options which would otherwise have to be considered. A "heuristic program" is a program employing heuristics to solve problems.

Any problem sufficiently structured to permit the identification of all possible solutions and having a means of comparing them can be solved by simply generating all the solutions and selecting the best one. Unfortunately, this is impractical for problems of any size. A classic example is the exploration of all paths through a checker game, which involves 10^{40} move choices and, at 3 moves per millimicrosecond, would take 10^{21} centuries to consider, (Samuel, [10]). Furthermore, many, if not most, important decisions relate to problems with insufficient structure to discern every solution or to yield an analytical means of comparing alternatives. Just as the human decision maker applies heuristics (rules of thumb) and intuition to determine a course of action under these circumstances, so must an automated decision process. It is important to note that heuristic methods do not guarantee optimal solutions, or, for that matter, any solutions at all. "All that can be said for a useful heuristic is that it offers good enough solutions most of the time"[3]. They do provide a capability to determine solutions to problems

[3] Feigenbaum, [1] p. 6.

which are either impractical to solve by rigorous methods, using algorithms, say, or for which no such methods are available.

Unlike its human counterpart who may develop decision rules over time from experience, the automated system must rely on the designer to specify a set of heuristics and when and how to apply them, (at least within the current state of understanding). However, it has been demonstrated[4] that they can sometimes be so defined as to permit internal modification which improves their efficiency and problem solving ability. The modification procedure is usually based on an analysis of past performance and, appropriately, is called "learning".

Learning is an important aspect of artificial intelligence and is exploited in the decision process developed later in this paper. An idea behind learning in problem solving programs is that to solve a new problem it is usually wise to try methods that have worked on similar problems in the past. Two approaches can be taken. One is to generalize on previous experience, learning by generalization, and the other is to retain past situations and retrieve them as needed to fit a problem at hand, rote learning. Rote learning has not proven very useful, apparently because of the storage and retrieval procedures required and the difficulty of resolving non-exact matching conflicts. Generalization has shown much promise and is used, in a specific form, in the development of the missile allocation decision process reported herein, so it will be discussed in some detail.

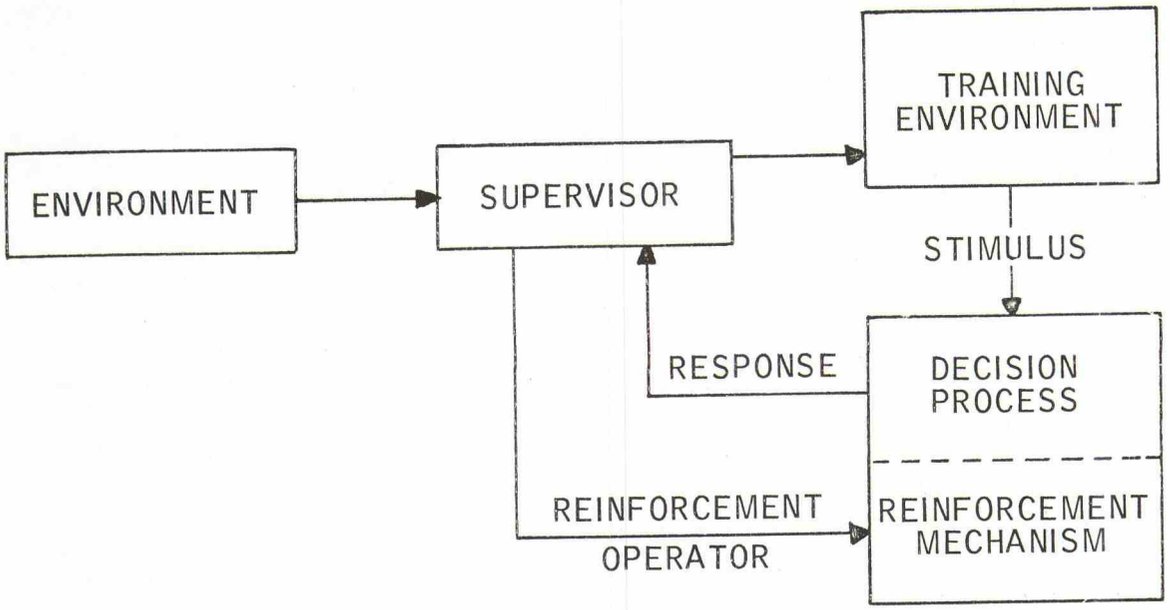
---

[4] See Samuel [10], Feigenbaum [2], Selfridge [12], Minsky [6].

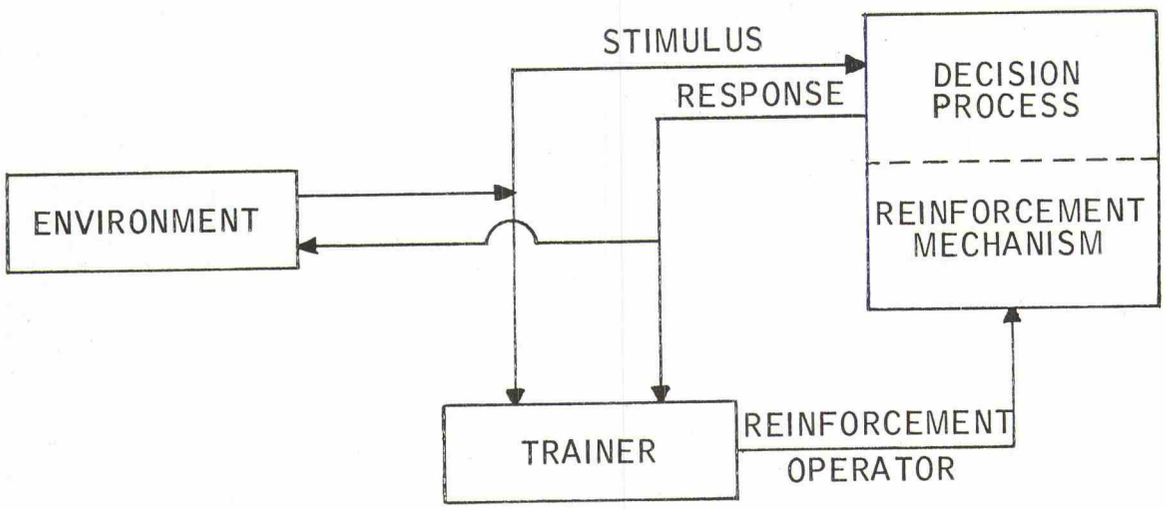
Suppose there are several elements in a decision procedure, each of which contribute to the outcome of a problem solving attempt. It is possible to generalize on the decision procedure by using a reinforcement operator which, if the outcome is "good" in some sense, will increase the prominence of those elements contributing in a positive fashion to the selection of the outcome and, if it is "bad", will do the same for those elements which contributed in a negative fashion. Theoretically, after repeated training sessions using the reinforcement operator, the decision procedure will have attained the proper correlation among its elements and thereafter provide consistently good solutions to new, but similar, problems.

A learning system to accomplish this result must contain a mechanism to trigger responses from the decision process and remember what role each element had in selecting the outcome, and a trainer to analyze the result and instigate appropriate reinforcement. Two systems are shown graphically in Figure 1.1. In both of these the same general flow prevails. An environment sends a stimulus to the decision process which makes one of several possible responses. The reinforcement mechanism remembers which elements entered into the outcome and how. The trainer analyzes the outcome and sends a reinforcement signal which increases or decreases the tendency to make the same decision to the same problem in the future. Note that the trainer need not know how to solve problems, but only to recognize good or bad outcome.

In the controlled learning system, (a), the supervisor acts as both the triggering mechanism, by establishing the training environment, and



(a) Controlled Learning System



(b) Operational Learning System

Figure 1.1 Two Systems for Reinforcement Learning

as the trainer. This system permits a maturation of the decision process under strict control to avoid instability, i.e., large, alternating changes in the correlation of elements, especially in the early stages of learning. The operational learning system, (b), utilizes stimuli from the real environment, which may be altered by responses from the decision process, and is subject to the whim of nature and the ability of the trainer to recognize good and bad responses.

There is a principle underlying difficulty in the design of decision processes with the ability to learn and that is the assignment of credit for reinforcement. Which elements are to be reinforced and by how much? In a complex decision process each element may have been applied many times in many different situations as sub-problems were solved or partial solutions generated. For example, in the game of chess the elements would be strategies and rules of thumb, e.g., "always check, it may be mate", which are applied hundreds of times, sometimes with success and sometimes with disaster, until finally, the game is won. To which elements go the credit?

One method of surmounting this difficulty that can be used with some decision processes is to reinforce at the subproblem level, (such as single move generation), where the effect of individual elements are more directly identifiable. The problem here is discerning good outcomes from bad, since the ultimate effect of a subproblem decision on the final outcome is generally not clear. Nonetheless, there is at least one example, the Samuel's Checker Playing Program, [10], where

this method has been employed with great success.[5] Briefly, this program evaluates a scoring polynomial, whose terms are properties based on the usual checker concepts, to measure the apparent goodness of each board position encountered. After an exchange of moves, the same board position is re-evaluated using the additional information available and the current and previous scores are compared. If the current score is greater than the previous score the coefficient of terms entering positively in the previous score are increased and those negatively are decreased. Opposite reinforcement is applied if the current score is less than the previous score. This learning technique for improving the board evaluation procedure has contributed significantly to the real time performance of this program which ranks with the top checker players in the country.

1.3 Artificial Intelligence and Operations Research

Heuristic programming is not a new concept in Operations Research. There currently exist heuristic programs which, to name a few, balance assembly lines (Tonge [15]), sequence jobs in job shops (Schwartz [11]), locate warehouses (Kuehn and Hamburger [5]) and solve the Travelling Salesman problem (Karge and Thompson [4]). These efforts could be construed as evidence that artificial intelligence is, in some sense, contributing to operations research. However the heuristic programs which solve operations research problems are limited to the systematic

[5] In fact, the Samuel's program was a pioneering work, and the most successful to date in sub-problem reinforcement.

application of static, albeit clever, heuristics and include none of the potentially powerful aspects of artificial intelligence, namely learning, planning, pattern recognition and induction. A possible exception is Tonge's Line Balancing Program, [15], in which a form of planning is used.

It has frequently been proposed that operations research is very powerful at the foreman level but of little use to the executive whose problems are not well structured. By the year 1957 the advance in the state of the art of artificial intelligence was so rapid that in November, speaking before a banquet at the 12th National Meeting of the Operations Research Society of America, H. A. Simon, himself a leader in the field, was prompted to say:

> "We are now poised for a great advance that will bring the digital computer and the tools of mathematics and the behavioral sciences to bear on the very core of managerial activity--on the exercise of judgement and intuition; on the process of making complex decisions... The way is now open to deal scientifically with ill-structured problems."[6]

Some progress has indeed been made. Tonge [14], reported on applying the Newell, Shaw, and Simon General Problem Solver (GPS)[7] successfully to the assembly line-balancing problem in 1963. And Clarkson, [0], built a model of a trust investment officer which closely matched the portfolio selections actually made by the individual

---

[6] Newell and Simon [9].

[7] The General Problem Solver is probably the most significant contribution to simulating human problem solving processes. Newell, Shaw, and Simon [7] and Newell and Simon [9].

in real situations. But, by and large, the use of artificial intelligence as a base for attacking ill-structured problems has not yet been extensively exploited.

1.4 Defense Missile Allocation: An Ill-Structured Operations Research Problem

At some point in the future, nations will have missile defense systems to counter nuclear armed, ballistic missile attack from their foes. Just when it will occur is an open question at this time, but there is little doubt that it will, in pace with technical advancement and political justification, and that the initial systems will employ non re-useable weapons, such as anti-ballistic missiles.

Suppose a missile defense system is operational; that is, the weapons and the detection, communication and control apparatus required to selectively employ them are in position and committed to the defense mission. At the moment hostilities commence, the design and configuration of the system become static. The location of defense missile sites cannot be changed. Neither can the number and type weapons assigned to these sites be changed. Every element is static except the defense strategy itself; how the available weapons are to be allocated among the incoming ballistic missiles.

The operations research problem, then, is to determine an optimal strategy (allocation procedure); optimal in the sense that it minimizes the total effect of the attack on the targets defended by the system. The realization of the strategy consists of discrete decisions (missile assignments) which are sequential over time and based only on information

available at decision time. Regardless of the structure of the function reflecting "total effect", at decision time the influence of decisions on this function is generally not measurable. For example, in Figure 1.2, if defense missile D2 is used against attacking missile A1 at time T, the probability that A1 will strike target A will be much lower than if A1 was unmolested by D2 and engaged by D1, since, if D2 was unsuccessful, D1 could still be useable against A1. However, by apparently reducing the real effect by thorough coverage of A1 at time T, it may well be that the opposite has occured as a result of depleting the D2 supply and thereby jeopardizing the defense of B and C from A2 and A3, respectively, at time T + t.

Any approach to this problem using existing techniques of mathematical programming under uncertainty must include a determination of the probability functions measuring the uncertainty. This is an unsurmountable obstacle, disregarding purely artificial specification. The recourse is heuristic programming.

1.5 Some Heuristic Approaches

A particularly simple heuristic program that is repeatedly used in military situations contains a single heuristic, "Shoot when you see the whites of their eyes." That is, engage whenever possible under the dictates of detection and weapon capability alone.[8] The advantage of

[8] Because of the prevalence of this approach in command situations, it is used as a base for measuring the relative effectiveness of the procedure developed in Chapter 3.

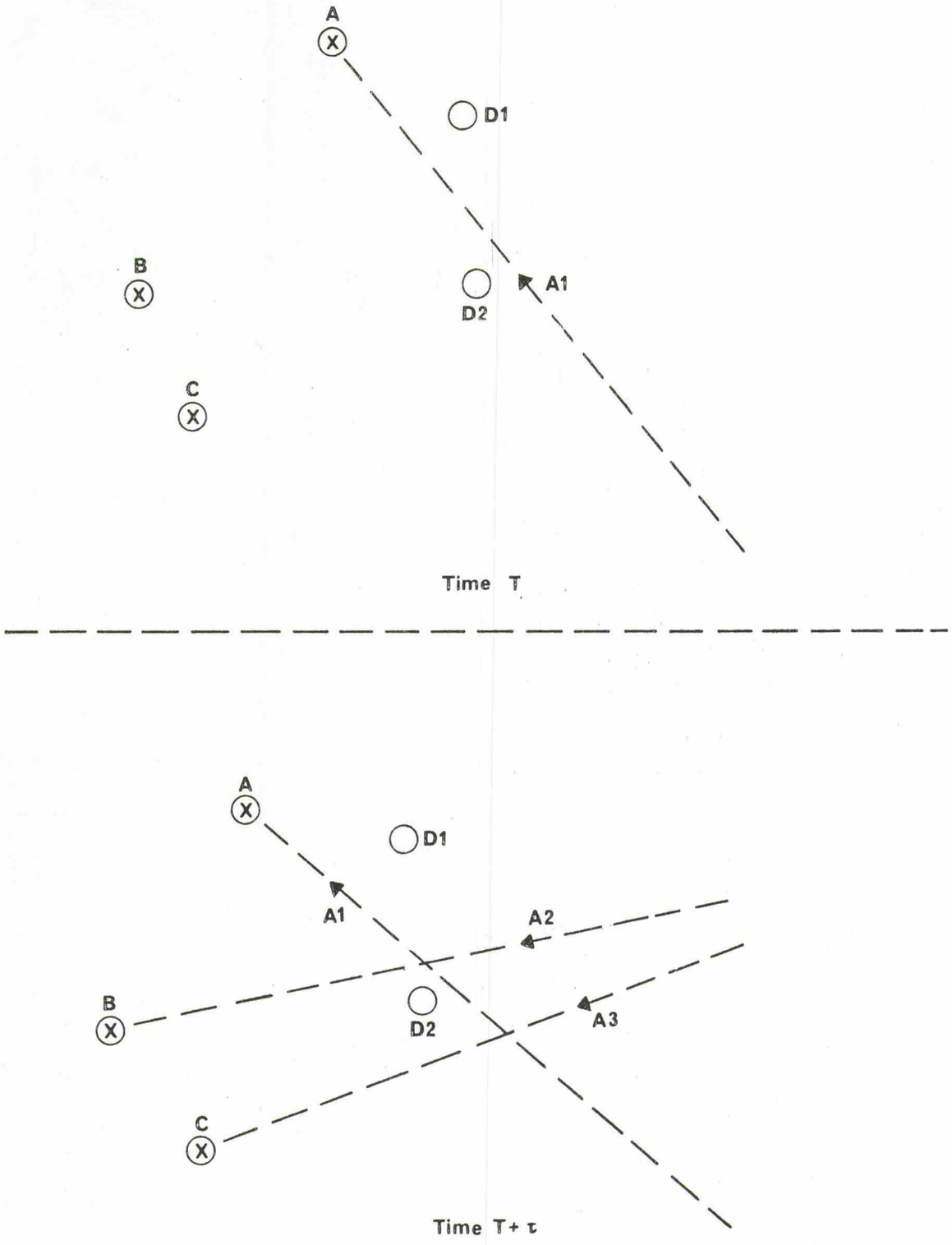


Figure 1.2 Typical Assignment Situation

this approach is that individual missile sites need not await confirmation from some central authority to engage a missile within their area, thus reducing reaction time and eliminating danger of noisy signals and communications failures. The price of this independence is the inflexibility of strategy which foregoes capitalizing on developing situations. Given sufficient numerical superiority in defense this approach might indeed be optimal, however it is difficult to visualize that circumstance.

Under the central control concept the typical heuristic program will attempt to conditionally combine rules of thumb about various aspects of the defense-attack environment to make allocations. The success of this type program depends entirely on the ability of the designer to discover good heuristics from human analysis of the problem. Now, if a method for permitting the program to adapt its decision process from its own analysis of the environment can be developed, the performance may be greatly improved. Possible ways of doing this are suggested by the research with game playing programs in artificial intelligence.

1.6 Game Playing Approach

The missile attack-defense situation is similar to two-person board game playing, checkers, chess, kalah, etc. In a board game, one player makes a move. The other considers the many moves he can make, weighs each in some fashion normally involving looking ahead (if I move there and he moves here and etc.) and finally selects a counter move. Then it is the first players turn, and so on. Each move is governed

by well defined, strictly enforced rules and effects the posture of both players for pursuing their goals, but the net influence of a single move on the outcome, win or lose, is obscure. As the game progresses, each player adapts his strategy to exploit weaknesses in his opponents play or to overcome deficiencies in his own. He may also adjust his procedure for selecting moves as the opponents strategy is identified or as phases of the games dictate, such as opening play or end game.

In defense-attack, the offensive party selects a strategy for destroying targets and initiates attack by launching ballistic missiles. Due to real problems in missile launching, location of missile launch sites and the attack strategy, the attacking missiles enter the defense domain sequentially over time. At some instant the defensive party observes those missiles within his domain (the offensive move), considers the possible responses and selects a specific missile allocation (the defensive move). On looking again, he observes new missiles (the next offensive move) as well as those remaining from previous moves, and the allocation process (move selection) is repeated. Meanwhile, the offensive party observes the effect of his strategy and, if possible, adjusts it to exploit weakened defense lines, compensate for weapon loss resulting in unaccomplished missions or otherwise improve his attack posture. Although no well-defined rules control the play, relevant actions are bounded by physical constraints, (weapon location, capability and numbers, target location, etc.) and causality laws, such as "destruction requires engagement."

Consider the following heuristic approach, suggested by game playing research in artificial intelligence directed toward understanding human decision making and learning processes, [10]. At each decision point, evaluate the resulting defense posture for each feasible missile allocation using a function of calculable "features" measuring critical (or at least useful) properties of the immediate environment, such as number of undefended targets, relative depletion rates among sites, expected loss, etc.; the evaluation should include looking ahead. Select the allocation yielding the "best" result. As the attack progresses, review past decisions by considering the additional information received since they were made. If the past decisions are not consistent, that is if the allocation selected at that time differs from the allocation which would have been selected had the subsequent information been available, then adjust the evaluation function so that it will be more likely to make the better decision if faced with the same problem again. The adjustment seeks to improve the discrimination among allocations and adapt to offensive strategy as it develops. Note that the evaluation function not only executes but also establishes defense strategy through the adjustment process.

Clearly, the overall performance of this approach is dependent on the particular features used in the evaluation function. However, the internal adaptability of the evaluation function provides, at least theoretically, the capacity to do the best possible under any given set of features.

The remainder of this paper is devoted to modeling the defense missile allocation problem and developing a heuristic program using the above approach.

II. THE ATTACK-DEFENSE ENVIRONMENT

2.1 A Missile Defense System

In this section a missile defense system is described which is technically consistent with current developments in electronic and space technology. No attempt has been made to design an optimal system and references to specific items of hardware are exemplary, only. The inclusion of the central control concept and area defense is dictated by the research goal, for without either one of them the capability for a flexible response is eliminated and, hence, so is the allocation problem as defined.[9]

A missile defense system consists of four functional subsystems and they are detection and tracking, engagement, control and communications.

The task of the detection and tracking subsystem is to detect attacking missiles as soon after launch as possible, well outside of the defense area, and actively track them, that is continuously determine their position, as they enter and progress through the defense area. A familiar example of a detection capability is the United States' Ballistic Missile Early Warning System (BMEWS), a chain of radar stations across the Arctic with the ability to detect intercontinental ballistic missiles

---

[9] Over the past three years articles, too numerous to identify individually, discussing the U.S. and U.S.S.R. activities in the missile defense area have appeared in Aviation and Space Weekly, Missiles and Space Digest and other national publications. An excellent overview of the U.S. Nike-X approach is G. A. Boehm's article, "Countdown for Nike-X", in the November 1965, issue of Fortune, p 133. He discusses both the system operation (the pictorial representation on page 134 of a local defense configuration is of particular interest) and the economic and political considerations.

approaching the Western Hemisphere from the Sino-Soviet land mass. The tracking mission requires sufficient numbers of tracking sites to provide multiple coverage on all incoming vehicles to avoid loss of acquisition and permit determination of exact trajectory, velocity, mass, etc. From this data, the impact point of the warhead can be calculated and the weapon yield estimated. A particular offensive tactic significantly effecting tracking is the use of decoys to confuse the defense system. Two methods are suggested in unclassified literature. The first is the use of non-productive warheads, i.e., warheads without a nuclear weapon, to cause the defender to waste his defense missiles and generally saturate the system. The second is to surround a productive warhead with a "cloud" of decoys so that it is difficult for the defender to identify and destroy the real weapon. In the system postulated here the resolution of the decoy problem is a responsibility of the detection and tracking subsystem. The output from this subsystem is clean data specifying the position, trajectory and impact point of each attacking missile in or approaching the defense area.

In military terminology, "to engage" means to intercept and destroy. It is the function of the engagement subsystem to actually destroy an attacking missile on command. Assuming the defense weapons are anti-ballistic missiles with nuclear warheads, destruction is accomplished by launching a weapon, guiding it within killing range of the hostile vehicle, and detonating the defense weapons' warhead. Defense missiles are located at missile sites, where several missiles are within close enough proximity of each other to be controlled by a single guidance device, say a ground based radar facility. There are two general

approaches pertaining to the engagement subsystem and they are local defense and area defense. For local defense, missile sites are located in the immediate vicinity of a specific target area and are equipped with short range, quick response weapons which can engage hostile vehicles attacking that area just prior to their reaching the target. Aside from enabling the concentration of defenses in particularly vital target complexes, like heavy population areas, this approach helps reduce the effect of the cloud decoy tactic. As the cloud re-enters the atmosphere the decoys decelerate more rapidly than the heavier nuclear warhead, which then can be identified and the quick response missile dispatched to destroy it. Area defense distributes missile sites strategically throughout the entire defense area to engage attacking missiles before they approach the target areas, thus contributing to the defense of many target areas from each site. With area defense, incoming missiles may pass within range of several sites thereby allowing for back-up if an engagement is unsuccessful, and for defense options on selecting a site from which to engage. Clearly, it is area defense that permits a flexible response, thus creating the need for an optimal allocation process.

The function of the control subsystem is to analyze all information relevant to the attack and initiate appropriate defense actions. Relevant information includes intelligence estimates of enemy capability (such as number and location of weapons, missile range and nuclear yield of warheads), possible attack strategies, current data provided by detection and tracking, defense system status, (such as number and capability of defense missiles at each site and the status of the targets within the

defense area), and specified defense strategies. Appropriate defense actions are to engage an attacking missile immediately from some particular site, to delay the engagement decision or not to engage at all. The defense system envisioned here has a central control to make all the allocations and initiate all engagements by communicating the necessary information to the selected missile sites.

The communication subsystem provides for the flow of information throughout the entire defense system.

The massive volume of data processed by each subsystem and the press of real time operation will require the use of very large, fast digital computers which can communicate directly with each other via core to core transfer. Data analysis and real time decision making will be automated. The human element will maintain command authority through man-machine communicating devices by imposing general strategy guidance or selecting from predesignated tactical options (such as identifying targets not to defend or missile sites from which to hold fire) but the bulk of command and control will be done by computer.

The targets within the defense area include population centers, industrial complexes, military installations, hydroelectric plants, communication facilities, transportation lines, bridges, port facilities and elements of the defense system itself. Each target has a designated type which identifies the principal resources located there, such as "population", "population/industrial", "strategic" and "defensive/strategic". Each target also has a number associated with it which is a measure of the value of the target with relation to the other targets. This value may vary either with time or defense strategy. A nuclear

retaliation installation would have a high value until all weapons were launched, and a much smaller one thereafter. A strategy concentrating on minimizing fatalities would place a much higher value on population centers than would a strategy for retaining military capability. Target value, as an adjustment parameter, plays an important role in the identification and implementation of defense plans by providing for a shift of emphasis between targets.

From the defense standpoint, the physical elements of the attacking force are the ballistic missiles of the enemy, initially as described by intelligence estimates of their location, range and payload and ultimately as they are detected and tracked within the defense area. Note that no overt action is taken by the defense system until the attacking missile physically enters its domain. In particular, the defense system described here does not include destruction of enemy vehicles at their launch sites although intelligence about pre-launch kills by friendly strategic forces can play a role in the defense decision process.

2.2 The Offense

A strategic nuclear attack is initiated to support national objectives and is intended to accomplish sufficient destruction of an adversary's national resources to permit the objectives to be realized. The overt action taken by the offense is to detonate nuclear warheads on selected targets using available delivery systems according to a plan designed to achieve the established goal with high probability. As with the defense, the offensive posture is completely established

when the attack commences, since the time span of interest precludes modification, and may only deteriorate from that point. Only the plan remains dynamic (at least theoretically) and is the instrument by which changes in objectives or strategic necessities are reflected.

The offensive plan is reflected in the type of targets attacked. For example, if the objective is to render ineffectual the military capability of a nation, the attack plan would certainly require placing weapons on military installations and transportation systems including ports, airfields and rail centers. Genocide, however, would initiate an attack on population centers and possible ignore the military installations except as they are co-located with population type targets. In general, an offensive plan will select certain types of targets and concentrate the attack on them. The term "type of attack" used in this paper refers to the type of targets under attack and is used as an indication of the attack plan being employed by the offense.

The delivery system of interest is the ballistic missile. The effects of other systems, (such as aircraft carried bombs, aircraft launched guided missiles and orbital vehicles) during an attack must be reflected in both the defensive and offensive tactics, but it is an implicit assumption that the missile defense system described in 2.1 is ineffectual against them; other defense measures are needed. The ballistic missile is positioned on a trajectory to the target during the powered phase of flight; thereafter, minor course deviation corrections may be accomplished by an onboard inertial guidance system. Depending on the effect desired, the warhead is detonated at an altitude above the target, on ground contact or underground.

2.3 Activity Flow

The overall flow of activity in an attack-defense situation can be described formally as follows.

(i) An enemy chooses an attack goal and launches ballistic missiles against applicable targets in accordance with a plan specifying timing, missile-target pairing, etc.

(ii) Attacking missiles (hereafter abbreviated AM) are detected by the defense radar network which calculates their trajectory and impact point and continuously communicates their position to the central control.

(iii) Control evaluates the data for each AM to determine its terminal point and which defense missiles (hereafter abbreviated DM) are capable of engaging it prior to reaching the terminal point. Note that a terminal point may not be one of the targets identified in the defense system.

(iv) Control analyzes available information on the immediate situation, expectations of future events and pre-specified defense strategies and decides which DM (if any) to allocate to each AM at the present time.

(v) The decisions to engage are communicated to the appropriate DM site and the DM is launched against the AM.

(vi) If the DM is successful or the AM reaches the terminal point, the AM is no longer in the system. AM's not destroyed advance along their trajectories, new AM's enter the system and the cycle returns to (ii).

(vii) As the attack develops, the enemy may either change to another attack strategy or adjust his plan to capitalize on weaknesses in the defense system, reallocate his missiles among remaining targets, etc.

III. THE PROGRAMMED MODEL - ACTS

3.1 Introduction

The method used to investigate the effectiveness of an adaptive defensive missile allocation process was to actually implement an automated central control and a training program to operate and guide it on the Burroughs B5500 Information Processing System. This program has the acronym ACTS taken from Automated Control with Training Supervisor. An overview of ACTS is depicted in Figure 3.1. The heart of the program is the Central Control System which utilizes a structured model of the attack-defense environment to interpret the action, and has a missile allocation procedure embodying the heuristic decision making and learning approach discussed in section 1.6. The Training Supervisor simulates the real world, including attacks against the defense system and results of engagements, monitors the allocation procedure and modifies the decision function to improve its performance. In this chapter the environment model is developed and definitions, heuristics and data files used by ACTS are presented. Finally the program, which is the realization of the model, is described in some detail. The learning process is explained in Chapter IV.

ACTS is programmed in the programming language "Burroughs Extended ALGOL for the B5500". A program listing and specifications are provided in appendix A.

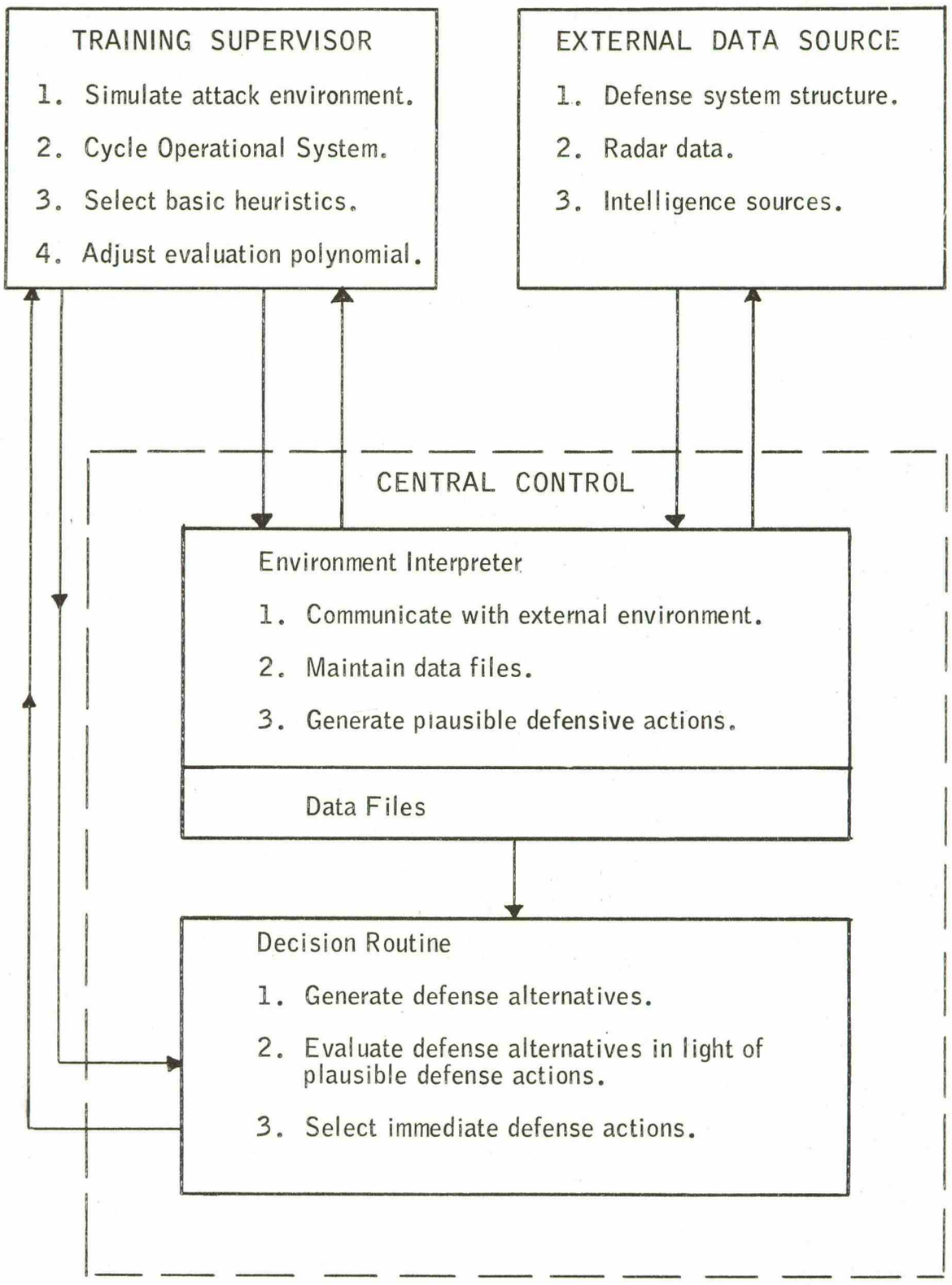


Figure 3.1 Functional Overview of Acts

3.2 Modeling the Environment for ACTS

Like most other real world situations the attack-defense environment cannot be represented exactly by a computer program, hence it is necessary to construct a model of it that can be so represented. The purpose of the model is to impose sufficient structure at a sacrifice of detail, yet maintain the essence and all critical features of the problem.

In the real world problem being modeled events occur in three continuous space dimensions and one continuous time dimension, i.e., attacking missiles enter the defense system at any instant of time and in any position in three dimensional physical space. Note that it does not seriously affect the analysis of the immediate situation, since positions are known and can be projected to any instant of time, but the continuous nature presents an infinite number of possible future events[10] for consideration. If an analysis of future events is to play a role in the decision process, this set must not only be made finite, but, in fact, quite small. To make it finite it is necessary to transform the continuous event space into a discrete space and to impose bounds on the space. To make it "quite small" requires further analysis and some good heuristics.

Consider first the vertical dimension in physical space. There are technical limitations on the height of an attacking missile. For one thing, the AM's are sub-orbital since orbital weapons are not

---

[10] An event is defined as any occurrence relevant to the attack-defense situation. Examples are an AM enters the system, a DM destroys an AM, a DM misses an AM, and an AM strikes a target.

relevant to the problem under discussion. Also, they are ballistic missiles which require a trajectory sufficiently high to carry them to the target. Hence there is a vertical band ranging from perhaps 50 to 100 miles above the earth in which all attacking missiles will be encountered. It is reasonable to assume that the designers of a defense missile consider this information so that every defense missile will have the capability for operating within this band. Hence, in the model developed here the vertical dimension is eliminated as a factor in the problem. An obvious alternative to complete elimination is classifying height as, say, high, medium and low, and categorize defense missiles by their operational floor and ceiling, but the effect is to increase the future event space by a factor of three for only a slight gain in generality so it was not adopted.

Transforming the remaining two dimensions is accomplished by laying a square grid across the geographical limits of the defense area and specifying the location of targets, defense missile sites and attacking missiles only by the square in which they appear. The accuracy of this representation is a function of the fineness of the grid. Every attacking missile can enter the system in one and only one of the grid squares and proceed to one and only one terminal square. Assuming that the selected grid square has dimension greater than zero and the attacking force is not infinite, under this model there are a finite number of square-to-square parings of one or more AM's. Further, since there is a finite number of DM's within the defense system there are a finite number of possible future events.

There remains the problem of continuous time, and it is relevant to both current activity and future events. From the standpoint of a control process, the time dimension is scarcely continuous, but is divided into sequential periods during which data analysis and decision making occur. At the beginning of each period the door is closed on incoming new data and the decision made from the information available up to that point. Data received during a period is not processed until the following period so the problem is unaltered if all data is considered as entering the system at discrete points in time, namely at the beginning of each period. To Control, this is equivalent to saying that events occur at discrete points in time, at the beginning of each period.

A time period may be either of fixed or variable length. Variable length time periods would apply to a mode of operation wherein the decision process is pre-determined so that the time required to arrive at an allocation is a function of the complexity of the immediate situation. If these time periods are not short, a problem arises in determining if a missile site will be within range of an AM when the decision is finally reached, since the AM is in fact moving during the decision process. Fixed length time periods avoid this complexity and permit the depth of analysis in the decision function to vary. The apparent paradox in fixed length periods is that the less complex problems are analyzed to a greater depth than the more complex ones. However, if the length is chosen to provide adequate time for sufficient analysis of most problems, the gain is that for many problems it is possible to do much more. The relative effectiveness of fixed, variable, variable with maximum and other approaches to the length of a period is not investigated in this research, but provides an interesting area for further study. The fixed length time period is adopted in this model.

A final complication in the real world environment which must be handled in the model is the response (lag) time of the engagement subsystem. Control decides to engage an AM with a particular DM and transmits the command. The DM must then be launched, proceed to engage the AM, the results noted (success or failure) and transmitted to Control. During the time required to accomplish this action Control is in doubt of the outcome of its previous decision. A likely heuristic would be to not consider allocating DM's to AM's which are in the process of being engaged. Except for the forward progress of the AM during the time of doubt, the resulting situation under the heuristic is equivalent to knowing the results of the engagement immediately. Since the exact location of an AM is not critical to the overall decision making power of the allocation procedure, and the timing mechanism and bookkeeping required to provide a simulation of lag time would significantly complicate the program, it was decided to assume that the results of a decision are immediate.

In summary, the model of the attack-defense environment used throughout ACTS is the activity flow given in Section 2.3 further constrained by dividing time into fixed length periods and reducing physical space for future events to a two dimensional square grid. All events occur at the beginning instant of the time period, the allocation decision is made and the results are available at the beginning of the next period.

3.3 Definitions, Heuristics and Data Files

In the remainder of this paper some terminology is used which may not be familiar to the reader and is now defined.

Active Target. Recall that every target is classified by the type of resource located there and that the type of attack employed by an adversary specifies which type targets will come under attack. An active target is a target with a type in jeopardy under the current type of attack.

Plausible Attack Corridor (PAC). Consider the geographical corridor formed by connecting a square on the periphery of the overlay grid and another square containing a target. If it is plausible for an adversary to attack a target square through such a corridor, then that corridor is a plausible attack corridor.

Active Plausible Attack Corridor. A plausible attack corridor leading to a square containing an active target is an active plausible attack corridor.

Ground Zero. Ground zero is the impact point of an attacking missile.

Attack Script. An attack script is the period by period specification of new information entering the system during a simulated attack. It includes AM data, newly assigned target values and changes in defense tactics.

Probability of Destruction. Associated with a defense missile the probability of destruction (PDDM) is the probability of the DM destroying an AM within range. Associated with an AM the term (PDAM) is the probability that the AM will destroy the target if it strikes it. If the AM is being engaged, the probability of destroying the target is $PDAM \times (1\text{-}PDDM)$.

Expected Target Value. For an individual target the expected target value is the multiple product of the current target value and the PDAM for each AM attacking the target.

Expected Value of an AM. The expected value of an AM is the expected value of the target it is attacking.

Covered. In reference to an AM, covered means that the AM can be engaged by at least one defense missile. In reference to a grid square or a plausible attack corridor, it means that at least one DM is effective in that area. In reference to a target it means that the target is locally defended.

Terminal Defense Missile (TDM). TDM is used in the program as synonymous with local defense missile.

Several parametrically specified heuristics have been included in ACTS to permit altering the defense tactics during the play. H4 applies only to the Training Supervisor and is used in the learning process; the others apply to the decision routine. All heuristics are initially inactive and must be overtly activated.

H1: Consider possible future events in seeking an allocation decision.

H2: If an AM arrives at a locally defended target during the current period, defend the target with the local defense weapon.

H3: Always engage an AM attacking a locally (a) undefended target and the AM is covered in only one or two more periods prior to reaching the target, or (b) defended target and the AM is covered by only one DM site or in only one period prior to reaching the target, unless engaging will endanger coverage of an active plausible corridor leading to an uncovered target. The rules (a) and (b) are extended to encompass (c) always engage if failure to engage forces, under (a) or (b), an engagement from the same DM site in a future period.

H4: In simulating the result of an engagement, use a probability of destruction of 1.

H5: Divide the AM into groups so that (1) all AM within a group are linked together by common DM which can engage them and (2) no AM can be

engaged by the same DM as an AM in a different group. Select allocation on each group independently. (See Section 3.5 for discussion of this technique.)

H6: Divide the AM into groups according to pre-specified DM site dependent criteria, such as geographical proximity of DM sites. DM site grouping must be input to table GROUPS as prescribed in Appendix B. Select allocation on each group independently. (See Section 3.5 for discussion of this technique.)

H7: Engage an AM whenever possible, (the "whites of their eyes" approach). If engagement is possible from more than one DM site, select the one with the most DM's remaining.

H8: Do not defend targets with a target value below a specified minimum. The minimum value, MINTGTVAL, must be given when H8 is used.

H9: Use only the calculated value of the decision polynomial to compare alternative allocations.

The data files used in ACTS are described in detail in appendix B, however, to facilitate the description of ACTS it is convenient to introduce them briefly at this point. Some data files define elements of the defense environment that are static in the pre-attack period and can be prepositioned in the program. Others are post attack files generated as the attack develops. Note that entries in the pre-positioned files may be modified in the post-attack period.

Prepositioned Data Files

DMSPEC: Range and effectiveness (probability of kill) specifications for each type DM in the system.

DMLOCO: Location of area defense sites with quantity by type of DM assigned.

TDLIST: Quantity of terminal defense missiles by TDM site.

TGTLCV: Location and value of each target in the system.

TGTTAD: Type, identification and covering TDM sites of each target in the system.

SQRDTA: For each grid square the area defense missile sites covering it.

SQRTGT: For each grid square the targets contained therein.

PACDTA: For each plausible attack corridor the entry and terminal grid square defining it, type targets contained in the terminal square, type targets not covered and an index to file PACDMS identifying DM sites covering the PAC.

ATTACK: For each AM in a simulated attack, the identification, time and location of detection, ground zero and speed.

Post-Attack Data Files

AMDATA: For each AM currently in the system, the AM and target identification, time period of arrival at target, and list of DM capable of engaging by time period.

ALTERS: Condensed version of AMDATA prepared for the decision routine to scan feasible alternative allocations.

TYPATK: A list of target types in jeopardy under the assumed type attack being employed.

3.4 The Decision Function

Recall from section 1.6 that the approach to optimal allocation of defense missiles is to compare feasible alternatives by evaluating, for each, a function (hereafter called the decision function) of the form

$$\Sigma_i \; W_i \; F_i$$

where the F_i are measures of relevant "features" of the defense environment and W_i are coefficients to be adjusted to seek optimality in the decision process. It is reasonable to expect the performance of a procedure based on this decision function to be sensitive to the particular features used. In the extreme it is possible to define features which have no bearing on the problem at all, such as counting the number of missile sites. Past use of this type of decision function has been in areas where there is a large amount of expert information on important aspects of the problem, namely checkers and chess. No expert information has been accumulated on the allocation of defense missiles, so the list of features defined below can no doubt be improved by additions, deletions, and modifications. However they seem to be adequate for demonstrating the feasibility of the approach, which is a primary goal of this research.

The overall objective of the defense system considered in the model is to minimize total loss in target value.

At each period the expected loss (risk) from the known AM can be measured for each feasible allocation; however, it does not reflect the risk from AM entering the system in the future. To consider all possible future events in adjusting current risks would be impractical because of the magnitude of the combinations it would entail. An indirect approach, which includes but is not determined by the current risk, is to calculate for each allocation the change in the defense posture, now defined as the ability of the defense system to protect the targets. Clearly, the defense posture may only deteriorate during an attack.

Hence, in seeking to minimize total loss in target value the decision process will in fact seek to minimize the rate of deterioration in the defense posture on a period by period basis.

Four characteristics of a good defense posture can be identified. They are maintain defense mobility, block attack routes, balance defense and avoid large expected loss. An area defense missile site may be considered as a highly mobile local defense facility since it protects many target areas. The extent of mobility for any particular DM site is proportional to the number of attack routes that it covers; a DM site covering six attack routes is more mobile than one covering only two. The overall mobility of the defense system is a function of the mobility of individual DM sites. Other things being equal, it is preferred to use a DM from the site covering the least attack routes, that is, maintain maximum mobility. The defense estimate of attack routes are the plausible attack corridors (PAC) defined above. Decision function features DMMOB1 and DMMOB2 reflect the degradation of mobility resulting from an allocation by penalizing, on a graduated basis, the use of DM from sites covering several PAC.

Clearly it is desirable to block attack routes by maintaining an area defense capability along each, otherwise an adversary can send AM to the target areas without fear of loss enroute. If the target area contains targets not locally defended the need for blocking attack routes is even more critical, especially if the undefended targets are active under the type of attack being employed. Decision function features CUND, CUNDA, CTUND and CTUNDA count uncovered PAC and penalize an allocation for opening attack routes. SOLEDM and SOLDMA count DM which alone provide coverage to one or more PAC.

Balancing the defense refers to avoiding uneven depletion among the DM sites. For example, in selecting between DM sites to engage an AM it is usually preferable to select the one with the most DM remaining, other

things being equal. An unbalanced defense permits an adversary to concentrate his attack through the weak points. Decision function features TLUND, TLUNDA, TLONE, TLONEA, DBALQ, DBALH, DBALT, and DBALA are measures of defense posture balance.

Avoiding large expected loss is the characteristic which relates directly to the overall objective. It implies the preferability of risking or actually assuming losses in order to prevent larger "expected" losses. Decision function features AMZERO, AMONE, and AMTWO measure aspects of expected loss.

A description of the features is given below. In ACTS the decision function is represented as a vector; the number on the left of the tag is the subscript of the feature in that vector.

1) CUND Number of undefended plausible attack corridors.

2) CUNDA Number of undefended active plausible attack corridors.

3) CTUND Number of undefended plausible attack corridors leading to locally undefended targets.

4) CTUNDA Number of undefended active plausible attack corridors leading to locally undefended active targets.

5) TLUND Number of locally undefended targets.

6) TLUNDA Number of locally undefended active targets.

7) TLONE Number of targets with one remaining local defense opportunity.

8) TLONEA Number of active targets with one remaining local defense opportunity.

9) AMZERO Expected value of AM facing no defense opportunity.

10) AMONE Expected value of AM facing one defense opportunity.

11) AMTWO Expected value of AM facing two defense opportunities.

12) DMMOB1 Number of applied DM covering from m to n PAC. (m and n are calculated for each defense environment so that one third of the DM sites cover m or fewer PAC, one third cover more than m but n or less and the remaining third cover more than n).

13) DMMOB2 Number of applied DM covering more than n PAC.

14) DMALQ Number of DM sites deploying one quarter of assigned missiles.

15) DBALH Number of DM sites deploying half of assigned missiles.

16) DBALT Number of DM sites deploying three quarters of assigned missiles.

17) DBALA Number of DM sites deploying all of assigned missiles.

18) SOLEDM Number of applied DM from DM sites which provide the sole coverage of a PAC.

19) SOLDMA Number of applied DM from DM sites which provide the sole coverage of an active PAC.

Coefficients, W_i, associated with the features are identified by the program subscript, that is, W1, W2, ... W19. Likewise, the features themselves are identified by F1, F2, etc. when use of the tags is not convenient.

As an option to comparing on the value of the function alone, ACTS permits lexicographic minimization of a threetuple (V1, V2, V3,) where V1 is CTUNDA, V3 is CTUND and V2 is the weighted sum of the features. The option is selected by system input.

3.5 A Search Reducing Technique

In a typical period attacking missiles will be transversing the defense system in many directions and in different geographical regions. If it is possible to divide them into independent groups, in the sense that the corresponding sets of covering DM are mutually exclusive, then the value of the decision function obtained by minimizing on each group separately and combining the result could be expected to be close to the minimum found by considering all AM collectively. It is not necessarily the minimum because, even though the groups are independent in the above sense, the DM are interrelated in the network of plausible attack corridors, however, the reduction in the number of feasible allocations to be evaluated is so dramatic as to make the technique attractive from the standpoint of processing time. For example, if twelve (12) AM are in the system and each has four (4) possible defense opportunities the total number of allocations is $4^{12} = \underline{16.8 \times 10^6}$. Now if two groups of six AM can be distinguished and processed separately, the number of allocations is only $4^6 + 4^6 = \underline{8192}$. Three groups of four AM yields $4^4 + 4^4 + 4^4 = \underline{768}$ and four groups of three yields $4^3 + 4^3 + 4^3 + 4^3 = \underline{256}$.

This search reduction technique is made available to ACTS by parametrically specifying either heuristic H5 or H6.

3.6 The Central Control System

The Central Control System contains two routines, the Environment Interpreter and the Decision Routine. Each is discussed as it would perform in an operational environment.

3.6.1 The Environment Interpreter

The function of the Environment Interpreter is to communicate with the outside world, interpret inputs in the light of the program model, maintain data files and relay allocation decisions to the engagement subsystem. The program flow follows.

E1: Accept data inputs and perform routine file update and maintenance at the beginning of each period, including

a) Number of DM and TDM available by site and type.

b) Specification of heuristics to be applied.

c) Specification of parameter for attack type determination procedure or type of attack directly.

d) Specification of target value.

e) Output control parameters.

E2: Process new AM data. For each AM entering the system generate an entry in AMDATA.

(i) From time, position, ground zero and speed calculate a trajectory projection.

(ii) Determine target, target value and local defense status.

(iii) Determine grid squares the projection passes through.

(iv) Locate candidate DM sites in SQRDTA.

(v) For each time period until the AM reaches ground zero, list the covering DM.

If there is no defense against an AM the value of the target is reset to an expected value and no entry is made in AMDATA. If the ground zero is not within the boundary of a target the AM is ignored.

If Heuristic H8 is active and the target value is less than MINTGTVAL the AM is ignored.

E3: Build ALTERS from AMDATA.

There is one row in ALTERS for each AM in the system. The first five entries in each row are bookkeeping indices for the Decision Routine, including the last relevant entry under heuristics H2 or H3. Following the indices are the DM covering, ordered by time period.

E4: If heuristic H5 or H6 is active and H7 is not, i.e., search reduction is to be used, determine independent AM groups and place the group designator for each AM in the final entry in ALTERS, otherwise set all final entries to 0.

E5: Go to Decision Routine. Return to E6.

E6: Relay DM engagement commands to appropriate facilities.

E7: Reflect the result of the periods activity in data files.

a) For AM striking a target, remove the AM from AMDATA and change the target value.

b) For AM destroyed by DM, remove the AM from AMDATA.

c) For DM deployed, remove the DM from DMLOCO.

E8: Go to E1.

3.6.2 The Decision Routine

The allocation of defense missiles is based on comparative evaluation of the decision function for each combination of assignments of a single

DM or TDM to each AM which is both feasible from the standpoint of availability and performance and permissible under specified heuristics. If future events are not to be considered by the decision routine, i.e., heuristic H1 not specified, then the particular combination having a minimum value of the decision function is selected. Ties are broken arbitrarily by selecting the first combination encountered yielding the minimum value. If future events are to be considered, H1 specified, then the value of each combination is modified by sending an AM through each plausible attack corridor (one corridor at a time) and selecting the covering DM minimizing the resultant decision function over all DM covering the corridor, holding the original combination fixed. The value assigned to the original combination is the maximum decision function value over all corridors. The combination with the minimum value is then selected. This forward looking procedure has the effect of looking at some of the adversaries future moves and expecting him to make the best choice (for him) under the decision function. Consideration of multiple AM, multiple PAC and looking forward more than one period is hampered by the additional time required to evaluate the large number of possibilities and is left for future investigation.

The imposition of defensive strategy on the decision routine is accomplished by the specification of heuristics. For example, heuristic H8 imposes the strategy of defending only those targets having an expected value above a given minimum, providing for the conservation of DM to defend high value targets. Such a strategy is useful in avoiding the deterioration of the defense in order to protect targets already destroyed or otherwise devalued, such as friendly stragetic missile sites after

missiles have been launched, as well as reflecting high level policy involving the sacrifice of certain targets for the general good. Other strategy heuristics included in ACTS are H2, H3 and H7.

By restricting the permissible combinations the strategy heuristics have the side effect of reducing search in the decision routine. Heuristics H5 and H6 are provided for the sole purpose of reducing search. Heuristics H1, the look ahead heuristic, and H9, the lexicographic heuristic, directly effect the evaluation process; however, H1 also increases search by a factor of the number of PAC. Note that if heuristic H9 is <u>not</u> active, the "value of the decision function" means the triplet (V1, V2, V3) as defined in section 3.4 and minimizing or maximizing is accomplished lexicographically; otherwise, only V2 is meant, or, equivalently, V1 = V3 = 0.

A brief program flow of the decision routine follows.

D1: If heuristic H7 is active then, for each AM covered in the current period, select a DM (or TDM) from the covering site with the most missiles remaining. Break times randomly. Go to E6.

D2: Set GRPNOW = 0.

D3: Evaluate the features for a feasible DM allocation for the AM with last entry in ALTERS equal to GRPNOW.

D4: Save the feature values in vector TFEAT and PFEAT. Set PACNOW to 1.

D5: If heuristic H1 is not active go to D9.

D6: For plausible attack corridor PACNOW, select the DM which minimizes the decision function given the allocation specified in D2.

D7: If the value of the decision function is greater than that for all previous PAC then save the feature values in PFEAT.

D8: If PACNOW < total number of PAC in the system, set PACNOW = PACNOW + 1 and go to D6.

D9: Using the feature values stored in PFEAT, if the value of the decision function is minimum over all previous allocations from D3, then save the allocation in vector BESTCOMB and PFEAT in vector PFEATURE. (Note that PFEATURE contains the feature values of the current minimum so it alone is considered to check for a new minimum.)

D10: Repeat D3 - D9 for all feasible allocations against group GRPNOW. On completion go to D11.

D11: If GRPNOW < total number of groups, set GRPNOW = GRPNOW + 1 and go to D3.

D12: The allocation minimizing PFEAT is in BESTCOMB. Those entries corresponding to current period engagements are stored in BESTALT and return is made to E6.

IV THE LEARNING PROCESS

4.1 Description

Figure 1.1 (a) in section 1.2 depicts a Controlled Learning System in which a training supervisor, responding to an external environment, established a training environment to send stimuli to trigger the decision process. The supervisor monitored the resultant decisions and reinforced the decision function to encourage good performance and discourage bad. The learning process in ACTS is such a system, with human input to the training supervisor portion of the program, and simulation and training thereafter controlled internally. It utilizes two methods of determining reinforcement actions, one of which provides a capability for learning in the operational environment, as shown in Figure 1.1 (b), Operational Learning System. Although operational learning has not been incorporated in ACTS, it is an immediate extension and is discussed later.

Reinforcement learning is essentially "hill climbing" on a surface defined over the set of all (normalized) weight vectors which reflects their relative problem solving ability. The goal is to arrive at the surface maximum. When the surface can be defined mathematically and has certain nice properties such as unimodality-continuity or convexity and the set over which it is defined is convex, methods exist for locating the surface maximum, either directly or by iterative search techniques which, at each point, determine a direction in which to move and how far to go. In the problem at hand the surface is essentially undefinable, can be expected to vary with each different attack plan and

have none of the above properties. However, in a particular situation a given set of weights will determine an allocation which can be compared with the best allocation for that situation and a single search iteration can be performed on the problem solving surface to adjust the weights toward the local maximum for that situation. An approach to subproblem reinforcement is to repeatedly apply this adjustment technique to many situations which together typify the total problem.

Given a typifying situation and a weight vector, the technique implemented in ACTS is composed of four steps.

1) Determine the key allocation for the situation. The key allocation is that allocation of DM to AM which is best in the given situation; it is not necessarily the one selected by the decision process.
2) Determine the direction in which to adjust the weights so as to move toward the (unknown) weights yielding the key allocation.
3) Determine the step size, i.e., how far to go in the direction specified in 2.
4) Adjust weights, initially as indicated by 2 and 3 and then to maintain logical consistency.

4.1.1 Determining the Key Allocation

In the controlled learning system the Training Supervisor (hereafter abbreviated TS) must determine the key allocation for each situation encountered. A particularly simple method, at least from the standpoint of the TS, is to provide the key allocation along with the situation. In the problem at hand, this implies that the human (external environment)

specifying the situation would also decide on the best allocation and include it as input to the TS. The main drawback with this approach is that the performance of the decision process is limited to that of the specifier. It is to be desired that an avenue be open for the decision process to excell its master, especially in an area where human experience is limited.

The alternative is to provide the TS with a capability for selecting the key allocation using some decision process differing from the one being trained. One such approach is to provide situations in logical, dependent sequences, such as a portion (or all) of an attack, so the TS may scan ahead and determine a "best" allocation in light of future requirements.

By having the TS scan to the end of an attack, the criterion for "best" could be reduced to that allocation in the current period permitting a defense allocation to the remainder of the attack which minimizes its total effect, this being the overall goal of the defense system.

Unfortunately, the number of allocations to be evaluated in such a deep look is astronomical. Typical training problems used in this research yield an order of 10^9 allocations for each four time periods.

An obvious compromise is to scan ahead only a limited number of periods and accept the allocation selected by the current decision function (considering all AM in those periods) as the key allocation. The underlying assumption in this procedure is that errors in judgment by the decision process in a single period are primarily due to lack of information as opposed to imbalance between terms in the decision

function, implying that reinforcement learning aims at improving generalization of current situations to future expectations. If the normalized weight vector being used by the decision process provides poor problem solving ability due to imbalance, it can be expected that the key allocation selected under the above procedure will be poor also, additional information notwithstanding. Experimentation has shown that retrogression, moving downhill, will generally occur at some point in the process by the selection of a key allocation which is inferior to the trial allocation. By aggravating the imbalance in the decision function, retrogression becomes self-generating and can be terminated only by intervention of the TS or by chance reversal. Bootstrapping in this manner from a weight vector with poor problem solving ability is inefficient compared with the method whereby correct key allocations are provided since each retrogression cycle must be compensated for. However, once the decision process attains a high level of ability, this method is quite successful in making minor adjustments, seeking a capability to respond to situations which are more complex than those previously encountered in the training cycles.

ACTS includes the capability to apply either method for selecting key allocations.

4.1.2 Determining the Direction

Given a trial allocation selected by the current decision function (with weights W_i) having feature values F_i and a key allocation determined by the TS with feature values F'_i, it is desired to adjust the weights W_i in the direction of W'_i, the (unknown) weights which

would have yielded the key allocation in this situation. Since the W'_i are unknown, it is necessary to find an approximating set W''_i, say, which has the property that it is possible to move from W_i toward W''_i and reach a point $\overline{W}$, say, such that using $\overline{W}$ will result in the preference by the decision function of the key allocation to the trial allocation. At this point it is convenient to introduce vector notation.

Let $W = (W_i) = (W_1, W_2, \ldots, W_n)$ be the current weight vector,

$W'' = (W''_i) = (W''_1, W''_2, \ldots, W''_n)$ be the approximation vector for the key allocation weights, W'_i,

$F = (F_i) = (F_1, F_2, \ldots, F_n)$ be the vector of feature weights for trial allocation,

$F' = (F'_i) = (F'_1, F'_2, \ldots, F'_n)$ be the vector of feature weights for the key allocation,

Then $D = W''-W$ is the direction in which the weights are to be adjusted.

Note first that

(i) $WF \leq WF'$

otherwise the decision process would have preferred the key allocation to the trial allocation.[14]

Now define the vector

$$F'' = (F''_i) = (F''_1, F''_2, \ldots, F''_n)$$

where F''_i is equal to:

---

[14] Lexicographic comparison is not used in the learning process as the weighted term V2 is dominated by V1 in the decision process.

$$
\begin{cases}
2 & \text{if } F_i = 0 \text{ and } F_i' < 0 \\
 & \text{or } F_i > 0 \text{ and } F_i' = 0 , \\
.5 & \text{if } F_i = 0 \text{ and } F_i' > 0 \\
 & \text{or } F_i < 0 \text{ and } F_i' = 0 , \\
F_i'/F_i & \text{if } F_i < 0 \text{ and } F_i' < 0 , \\
F_i'/(F_i'-F_i) & \text{if } F_i < 0 \text{ and } F_i' > 0 , \\
(F_i-F_i')/F_i & \text{if } F_i > 0 \text{ and } F_i' < 0 , \\
F_i/F_i' & \text{if } F_i > 0 \text{ and } F_i' > 0 , \\
1 & \text{if } F_i = 0 \text{ and } F_i' = 0 .
\end{cases}
$$

Clearly

$$
\begin{aligned}
& F_i'' > 1 \quad \text{if} \quad F_i' < F_i , \\
\text{(ii)} \qquad & F_i'' = 1 \quad \text{if} \quad F_i' = F_i , \text{ and} \\
& 0 < F_i'' < 1 \quad \text{if} \quad F_i' > F_i .
\end{aligned}
$$

Finally let the approximation vector elements

$$W_i'' = W_i F_i'' .$$

From the relations (ii), the following relations hold.

$$
\begin{aligned}
\text{(iii)} \qquad & W_i'' > W_i \quad \text{if} \quad F_i' < F_i \quad (F_i'-F_i < 0) \\
& W_i'' = W_i \quad \text{if} \quad F_i' = F_i \quad (F_i'-F_i = 0) \\
& W_i'' < W_i \quad \text{if} \quad F_i' > F_i \quad (F_i'-F_i > 0)
\end{aligned}
$$

and $W_i'' > 0$ since $W_i > 0$ and $F_i'' > 0$.

It remains to show that under the approximation W'' for W', the direction $D = W''-W$ has the desired property, namely that it is possible to move from W in the direction D and arrive at a new vector $\overline{W}$ such that by using $\overline{W}$ the decision function will prefer the key allocation to the trial allocation, that is $\overline{W}F' < \overline{W}F$.
A direction with this property will be called a <u>feasible direction.</u>
Note first that from (iii)

$$W''(F'-F) < W(F'-F)$$

$$W''(F'-F) - W(F'-F) < 0$$

$$(W''-W)(F'-F) < 0$$

$$D(F'-F) < 0$$

hence,

(iv) $\quad DF' < DF$.

Define $\overline{W} = W+SD$ where $S \geq 0$ is a scalar. It can be shown (see Appendix F for proof) that

Lemma: For any fixed, finite W, D, F and F', there exists a finite scalar $S^* \geq 0$ such that for any $S > S^*$,

$$\overline{W}F' < \overline{W}F .$$

Hence, conclude that $D = W''-W$ is a feasible direction.

4.1.3 Determining the Step Size

Having found a feasible direction D in which to move, it is necessary to determine the step size, S. A seemingly logical approach, and one that was initially applied, is to calculate S^* as in the Lemma, see Appendix F, equation (v), and use it as a step size, thereby guaranteeing that under $\overline{W}$ the decision function will at least be neutral between the trial allocation and the key allocation. However, this approach led to severe instability in the learning process. In the first place, the situations considered only provide approximations to the problem solving surface, hence the key allocation itself is at best an approximation. Secondly, D is based on an approximation, W'', to weights W' yielding the key allocation; there is no guarantee that $\overline{W}F' \leq \overline{W}F^k$ for all k, i.e., for the feature values of all other feasible combinations. Finally, S^* does not consider the past performance of the decision function and hence does not permit dampening of reinforcement with maturation of the decision process.

To offset the instability, a conservative approach has been adopted in which the step size, S, is determined as the minimum of:

(a) The reciprocal of a maximum feature weight repeat factor, R_i, calculated as follows for each feature weight, W_i:

After each learning situation if W_i was unchanged, then $R_i = R_i + 1$ ($\max R_i = 10$); otherwise $R_i = (\text{largest integer of } (R_i/2))+1$,

or

(b) The quantity $(WF'-WF)/WF'$

but in no case less than .05 .

(Initially a minimum of .1 was used but as performance improved, it became useful to reduce it to .05 .)

A modification of this approach was also used with reasonable success, namely to apply (a) above to calculate an S_i for each individual feature weight and adjust individual W_i by a weight of S_i . This procedure enables a more rapid modification of W_i with a poor performance record as indicated by frequent modification, or of new features being introduced into the decision process. The effect of this is to tilt the direction vector, D, in the W_i plane. However, if S_{min} is the minimum S_i and $S_i^* = S_i/S_{min} \geq 1$ then by modifying W_i'' and W_i such that $W_i^{**} = S_i^* W_i'' = S_i^* W_i F_i'' = W_i^* F_i^*$, the relations (iii) and all subsequent mathematics hold with W** and W* replacing W" and W, respectively, and the new direction is also feasible. The step size is set at S_{min} as before.

4.1.4 Making Weight Adjustments

The final step in the reinforcement learning technique in ACTS is to make the weight adjustments. After calculating $\overline{W} = W + SD$, as indicated above, certain logical operators are applied to the elements of $\overline{W}$ in order to retain consistency within the decision function. For example, consider the two features DMMOB1, the number of DM deployed from sites covering m to n PAC, and DMMOB2, the number of DM deployed from sites covering more than n PAC . Clearly, the deterioration of the defense considering mobility alone is greater if a DMMOB2 DM is used than if a DMMOB1 DM is used, all else being equal.

Hence, the weight associated with DMMOB1 should be not greater than the weight associated with DMMOB2. If in $\bar{W}$ this consistency has been destroyed it is recovered by adjusting the subject weights as necessary. There are currently twenty consistency analyses in the Training Supervisor.

The resultant $\bar{W}$ then replaces W and the process cycles through the next situation.

4.2 Flow of the Training Supervisor

In addition to supervising the reinforcement procedure, the TS is responsible for establishing the training environment and simulating results from decisions and attack scripts. A flow of the TS follows:

T1: Read inputs (punched card) and establish training environment. The types of data input and resulting data files are:

a) Target description for each target in the defense area: TGTLCV and TGTTAD are generated.

b) Defense missile specifications. (range, probability of kill): DMSPEC is generated.

c) Defense missile site data. (location, number DM by type): DMLOCO is generated.

d) Plausible attack corridor definition (entry and terminal grid square): PACDTA and PACDMS are generated.

e) AM portion of the attack script: ATTACK is generated.

f) Heuristics for activation: activate appropriate heuristics.

g) Simulation and learning instructions (number of times to cycle the attack script, whether or not learning is to be

accomplished and, if so, what parameters to apply):
set CYCLE, LEARNING and REPEAT, or BOOKGAME.

h) Feature weights for initial use: vector WATE is set.

i) Run identification: store in RUNID.

Additional inputs to modify the environment or alter the run parameters may be provided that are identified by the period in which they are to enter the system. These inputs are read directly by the Environment Interpreter.

T2: Step the Environment Interpreter through E1, E2, E3 and E4, setting up AMDATA and ALTERS for AM relevant to the current period.

T3: If not in a learning mode, step through E5 (make DM allocation) and E6. Go to T15.

T4: If key allocation is provided then go to T17 else go to T5.

T5: Recycle E2, E3 and E4 to bring in the AM entering during the next period.

T6: Step D2.

T7: Step D3 through D9 to evaluate features for a particular allocation against AM currently in the system, not including those entered in T5, and determine minimum valued allocation thus far.

T8: Evaluate the features for a feasible DM allocation for the next period AM entered in step T5, given that the DM allocation against current AM is fixed at the last D3 combination. Store the resulting feature values in LFEAT.

T9: Using the feature values stored in LFEAT, if the value of the decision function is minimum over all previous T8 allocations given this D3 allocation, then store LFEAT in LPFEAT. (Note that LPFEAT contains the feature values of the current minimum under this (fixed) D3 allocation.)

T10: Repeat T8-T9 for all feasible allocations with this D3 allocation fixed.

T11: If the value of the decision function using the features in LPFEAT is minimum over all previous D3-T8 combinations, then store the allocation in LBESTCOMB, LPFEAT in LFEATURE and PFEAT in LPFEATURE (note that LPFEATURE contains the PFEAT corresponding to the D3 combination (stored on LBESTCOMB) yielding the best overall allocation so far considering a look ahead to the next period).

T12: Repeat T7 for all feasible D3 combinations. On completion go to T13.

T13: If the D3 combination chosen in D9 is not the same as that chosen in T10, assume the decision procedure has made an error since if it had more information (namely the next period attack pattern) it would have made a different selection. Hence, reinforce the feature coefficients (WATES) so as to make it more likely to make the "correct" decision in the same situation. The reinforcement method is as defined in sections 4.1.1 through 4.1.4 with $F \equiv$ PFEATURE and $F' \equiv$ LPFEATURE. Adjust the weight repeats R_i as indicated in 4.1.4.

T14: Repeat steps T6 through T13 until the same D3 combination is selected by D9 and T10 or the number of repetitions is equal to REPEAT (a control parameter entered in T1, g).

T15: Simulate the outcome of DM engagements (using the D1 or D9 allocation) using random numbers and kill probabilities. (Note that if H4 is active the kill probabilities for every DM and TDM is taken to be 1 . This can be used to eliminate chance variations in results while comparing the performance of specific decision procedures or WATE settings.)

T16: Step E7 to update files and go to T2.

T17: Read in key allocation.

T18: Step D3 through D9 to evaluate feature for a particular allocation against AM currently in the system. If this allocation is the key allocation, store the feature values in LPFEATURE. If this allocation has the minimum evaluation so far, store the feature values in PFEATURE and the allocation in BESTCOMB.

T19: Repeat T18 for each feasible allocation then go to T20.

T20: If the allocation selected by the decision function is not the same as the key allocation and the value of the decision function for the two allocations are not equal, then reinforce the feature weights (WATE) using the method defined in sections 4.1.1 through 4.1.4 with

F ≡ PFEATURE and F' ≡ LPFEATURE

T21: Adjust the weight repeats R_i as indicated in section 4.1.4.

T22: Repeat steps T18 through T21 until either the key allocation is selected by the decision process or the number of repetitions is equal to REPEAT (a control parameter entered in T1, g).

T23: Go to T15.

V IMPLEMENTATION

5.1 Introduction

The general methodology for developing a self-modifying decision procedure for the allocation of defense missiles was determined from an analysis of the overall missile defense situation and prior research in the field of artificial intelligence. However, much of the specific material in Chapters III and IV evolved from a continuing experiment in which simulated missile attacks are inflicted on defense environments for the purpose of either decision function modification or analysis of the effectiveness of the decision process. Within this experiment lies the justification for the techniques employed in the model, ACTS, and the verification of the results reported below. Principal difficulties in the design of the experiment stem from the nonavailability of real data and recognized expert missile defense decision makers, a result of the futuristic nature of the problem. Circumvention of these deficiencies within the scope of the stated research objectives (section 1.1) is claimed on the basis of selecting plausible, nonrestrictive environments and noting that, with the growth of a missile defense expertise, it is expected that the decision function could be further modified to give improved performance, but this would not invalidate the results so far achieved.

In this chapter the experiment is described and a summary of the more significant developments is presented. The general research technique has been to cycle the learning process and then check the new decision function against past performance. Each cycle is analyzed to

ascertain cause for errors in judgment, retrogressive learning or other critical performance factors and, if necessary, adjustments are made to compensate for particularly difficult decision problems, e.g. a new training attack may be added to cover a situation previously omitted or a feature added to the decision function to cover a "blind spot". In the interest of demonstrating the feasibility of applying heuristic learning techniques to the problem, these external adjustments specifically excluded direct modification to the weight vector. The internal learning process alone is responsible for the evolution of the weight values.

5.2 Establishing the Experimental Environment

Under the model of the attack-defense environment presented in section 3.2, establishing an experimental environment requires specification of a target system, defense sites, missile capability, plausible attack corridors and attack script. In order to avoid the hazard of biasedness present in creating a purely artificial data base, the western portion of the United States was selected as a general pattern for the target system. This segment provides both congested metropolitan areas and large expanses where targets are sparse, and the coastal region allows for late detection and shallow defense situations. No attempt was made to include all identifiable targets nor to accurately locate those which were included. For convenience the coordinate system was rotated approximately ninety degrees. Target types are identified as the single letters P, D, S, M, I or some combination thereof, and target identification is simply the target type followed by a sequence number, e.g. P11, P12, DS7.

A system of defense sites is distributed throughout the target system which provides for overlapping coverage around most of the perimeter of the defense area, under the assumptions on DM range, and some coverage in depth along principal attack routes. Terminal defense sites were assigned to targets somewhat arbitrarily but, in general, on the basis of target value. The number of TDM and number and type of DM assigned to each site is specified with each attack script; however, the location of the sites remained fixed during the entire experiment. Figure 5.1 shows the targets and DM sites and Appendix E contains a listing of the target and defense missile site data. Missile performance specifications were fixed at one unit range and probability of kill of .75 for DM and probability of kill of .95 for TDM.

From a reasonable analysis of attack possibilities and adversaries goals, a set of attacks were designed for training and decision analysis. Initially, a single large attack on selected types of targets throughout the entire defense system was used to determine the effectiveness of both the decision process and learning technique. Subsets of this attack were used as training attacks for the learning process. For greater flexibility and efficiency, the large attack approach was downgraded in favor of smaller attacks on portions of the defense system, each of which emphasized a specific set of attack situations. Appendix D contains a summary of the attacks.

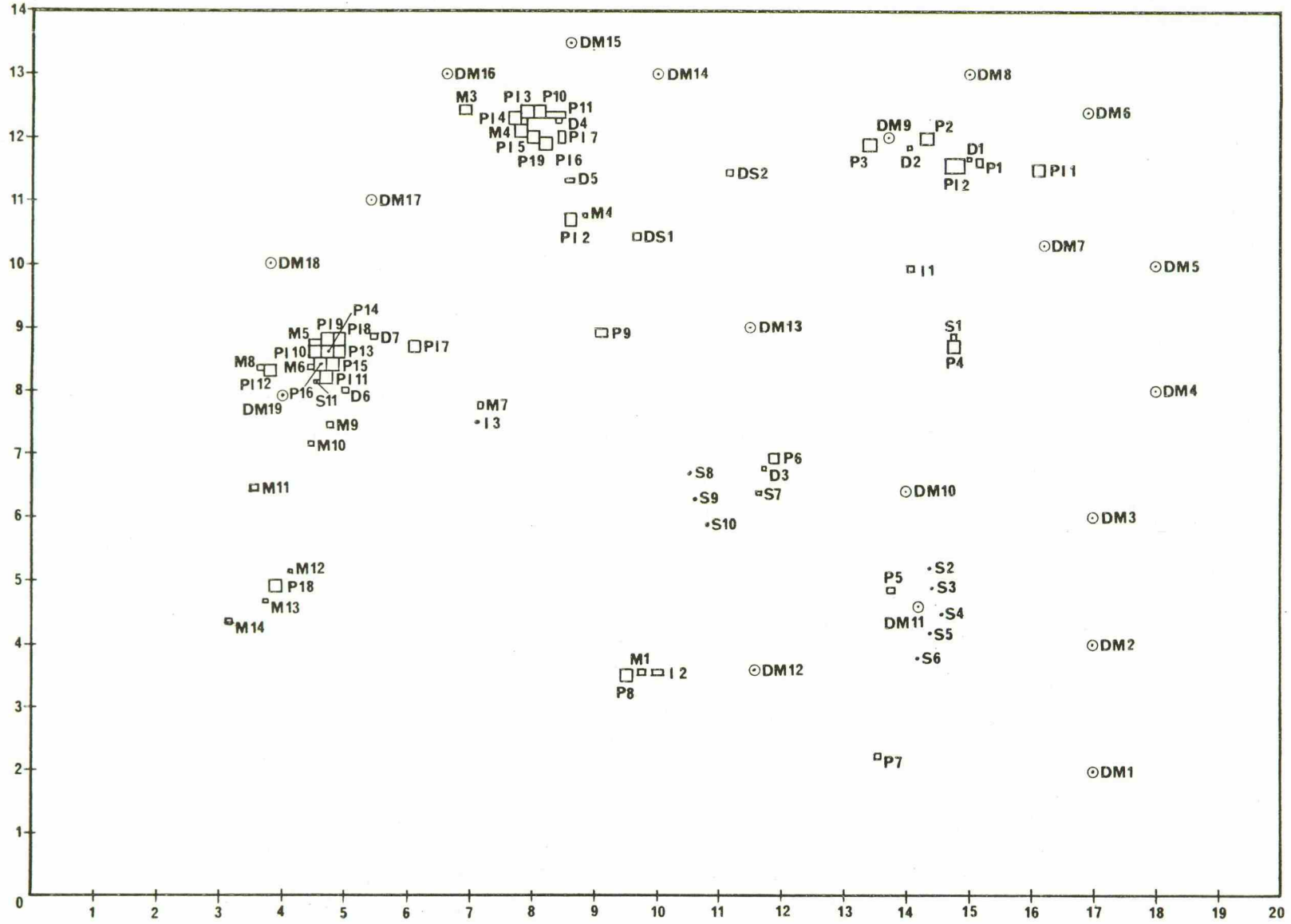


Figure 5.1 Test Environment Target and Area Defense Missile System

5.3 Evolution of the Critical Features

The selection of a set of critical features which are responsive to the important aspects of an ill structured decision problem is an evolutionary process. An initial set can be determined from problem analysis or analysis of human decision making techniques; however, it can be expected to be deficient due to oversight and the difficulty of translating human decision processes into analytical decision processes. Part of the evolution is to adjust for these omissions and misrepresentations.

The features initially included in the ACTS decision function were designed to reflect the four characteristics of good defense described in section 3.4, namely mobility, block attack routes, balance and avoid catastrophe. Specifically, these features were CUND, CUNDA, CTUND, CTUNDA, TLUND, TLUNDA, TLONE, TLONEA, DBALQ, DBALH, DBALT, AMZERO, AMONE, AMTWO, DMMOB1, DMMOB2, UNCSQR AND RATIO. UNCSQR was a count of the target system squares which were not covered by a least one DM and was applied to balancing the defense posture. RATIO was defined as the sum over all AM in the system of the expected value of the AM divided by the number of remaining defense opportunities against the AM if the selected defense was unsuccessful. RATIO was applied to avoiding large losses.

UNCSQR was deleted after several learning runs failed to change its weight and analysis disclosed that the decision function was insensitive to its value in the sense that its absence from the decision function would not change any decision made with it in. RATIO was deleted after nineteen learning runs because its weight was converging

to zero, indicating that it was consistently contributing to the selection of other than the key allocation.

The absence of DBALA (number of DM sites voiding) was a deliberate omission based on the expectation that the effect of a DM site voiding would be adequately reflected in CUND, CUNDA, CTUND, CTUNDA and UNCSQR. However, allocation errors were being made by voiding DM sites without increasing any of these values; in fact, pathological situations arose in which voiding a DM site actually reduced the value of the decision function by reducing only the value of DBALT. DBALA was added and significantly improved the decision making quality.

The final alteration of the critical features was the addition of SOLEDM and SOLDMA. The decision process was penalizing allocations which left attack corridors undefended, but was not constraining successive allocations leading to defense situations placing the attack corridors in jeopardy of being voided. The DM sites covering an attack corridor would be depleted until a single DM remained before the decision process restricted further depletion (by CUND, CUNDA, CTUND and CTUNDA), but by then, the posture was deteriorated and there was no defense against further weapons transversing the corridor. As a result, the performance of the decision process was handicapped and, at the same time, the learning process was unable to compensate for this type decision error since it had no features which reflected the difficulty. Analysis showed that if a penalty could be imposed for using a DM from a DM site which alone was covering an attack corridor, many of the situations could be avoided, hence, SOLEDM and SOLEDMA were added. Clearly, this approach could have been extended to "only two" (or three

or etc.) DM sites covering a corridor, but the present performance is adequate without further depth in this area and no positive improvement from their inclusion is apparent.

5.4 Evolution of the Learning Process

The learning process currently applied by ACTS was developed over the course of experimentation. Initially, a naive learning scheme was employed which consisted of running a large attack against the entire target system and, at each period, permitting the training supervisor to select a key allocation from looking into the future. A full step size was used in weight adjustment and, if an adjustment was made in a period, that period was repeated until either no adjustment was required (i.e., key allocation was selected) or a repeat parameter was reached, usually two or three repeats. The results were unsatisfactory. Not only was the procedure inefficient from the standpoint of search per adjustment, but it was also highly unstable in the sense that neither the weights converged nor the performance improved. In fact, the performance deteriorated. Two factors were held as the primary cause, the generality of the large attack and the lack of dampening the step size with correct decisions. The first disallowed for phased learning in which the decision function is balanced toward one type situation (e.g. mid-attack, voiding DM's, voiding TDM's) before it must be adjusted for a significantly different one. The second permitted full adjustment of a weight regardless of its past decision making contributions, aggravating retrogression and instability whenever incorrect key allocations were selected by the TS.

Following this abortive attempt, the large general attack was abandoned in favor of smaller attacks emphasizing particular situations, a step dampener was added to the learning process, each weight was reset to a value of one (1), and the learning process reinitialized. Six training attacks were established and they are:

TA1 emphasizing mobility and balance,

TA2 emphasizing PAC coverage,

TA3 emphasizing expected loss tradeoff,

TA4 emphasizing the terminal phase of an attack through the PACs,

TA5 emphasizing the terminal phase of an attack not through the PACs,

TA6 emphasizing the intermediate phase of an attack.

The training approach is to cycle a complete training attack through the learning process until either the decision procedure selects the key allocation every period or a cycle limiting parameter is reached (usually set at 5 or 10), then a different training attack is selected, and so forth.

Table 5.1 and Figure 5.2 and 5.3 depict selected sets of feature weights ensuing from this approach and Table 5.2 lists the cumulative number of cycles and attacks for each set. The weights shown are normalized for display by dividing each raw weight by one quarter of the sum of the raw weights. Until the entry of DBALA at set 3 and SOLDMA and SOLEDM at set 4, the corresponding weights are taken as zero and not shown on the graph. Note that the scale in Figure 5.2 is half that of 5.3 in order to show the AMZERO weight. The DBALA weight is included on both to provide continuity.

NORMALIZED FEATURE WEIGHTS

SET	CUND	CUNDA	CTUND	CTUNDA	TLUND	TLUNDA	TLUNE	TLUNEA	AMZERO	AMUNE
1	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500
2	0.0424	0.0847	0.0424	0.1694	0.6777	0.3388	0.1694	0.0847	1.3553	0.0188
3	0.1758	0.1758	0.1758	0.1758	0.3516	0.3516	0.1758	0.1758	0.7033	0.1758
4	0.0217	0.0708	0.0318	0.1040	0.2860	0.2860	0.0953	0.1129	1.2892	0.0898
5	0.1190	0.3044	0.1190	0.3044	0.2450	0.2821	0.1251	0.1253	0.7394	0.0324
6	0.0201	0.2606	0.0224	0.2909	0.1647	0.3138	0.0603	0.0603	0.9999	0.0334
7	0.0161	0.2095	0.0210	0.2728	0.1775	0.3323	0.0623	0.0656	1.2359	0.0239
8	0.0175	0.2277	0.0220	0.2857	0.1845	0.3335	0.0580	0.0644	1.2213	0.0205
9	0.0155	0.2014	0.0227	0.2948	0.1891	0.3297	0.0586	0.0685	1.3081	0.0165
10	0.0151	0.1954	0.0244	0.3169	0.1834	0.3087	0.0621	0.0752	1.3298	0.0146
11	0.0153	0.1988	0.0237	0.3071	0.1777	0.2992	0.0630	0.0763	1.3507	0.0141
12	0.0149	0.1932	0.0230	0.2984	0.1727	0.2907	0.0642	0.0777	1.3756	0.0137
13	0.0146	0.1893	0.0220	0.2856	0.1736	0.2922	0.0612	0.0748	1.3791	0.0131

NORMALIZED WEIGHTS CONTINUED

SET	AMTWO	DMMOB1	DMMOB2	DBALQ	DBALH	DBALT	DBALA	SULEDM	SOLDMA
1	0.2500	0.2500	0.2500	0.2500	0.2500	0.2500	0.0000	0.0000	0.0000
2	0.0424	0.3812	0.1694	0.0847	0.1694	0.1694	0.0000	0.0000	0.0000
3	0.0440	0.3516	0.3516	0.0879	0.1758	0.1758	0.1758	0.0000	0.0000
4	0.0087	0.0096	0.3096	0.0029	0.0308	0.0963	0.8520	0.0241	0.2785
5	0.0155	0.3229	0.4050	0.0303	0.0835	0.1612	0.3029	0.0722	0.2103
6	0.0076	0.3368	0.4212	0.0278	0.0607	0.1762	0.5727	0.0103	0.1603
7	0.0080	0.3135	0.3629	0.0222	0.0744	0.1355	0.4923	0.0087	0.1654
8	0.0082	0.3026	0.3854	0.0192	0.0644	0.1173	0.5310	0.0076	0.1292
9	0.0091	0.2945	0.3563	0.0164	0.0550	0.1001	0.5362	0.0065	0.1212
10	0.0096	0.2856	0.3456	0.0145	0.0488	0.0889	0.5680	0.0057	0.1077
11	0.0095	0.2906	0.3423	0.0141	0.0473	0.0861	0.5769	0.0056	0.1017
12	0.0097	0.2824	0.3486	0.0137	0.0460	0.0837	0.5875	0.0054	0.0989
13	0.0102	0.2905	0.3490	0.0129	0.0484	0.0781	0.6034	0.0052	0.0969

TABLE 5.1

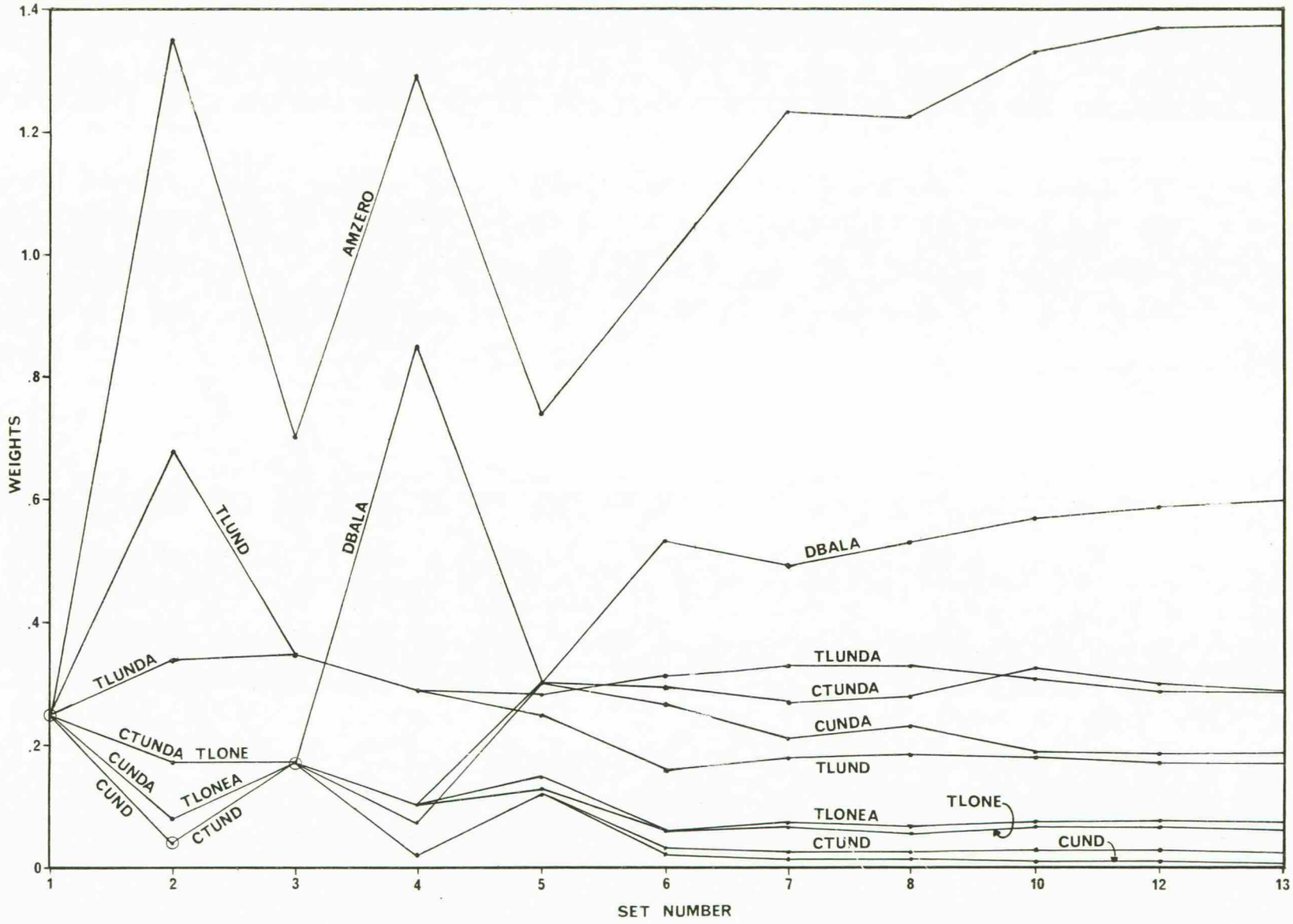


Figure 5.2 Normalized Feature Weights

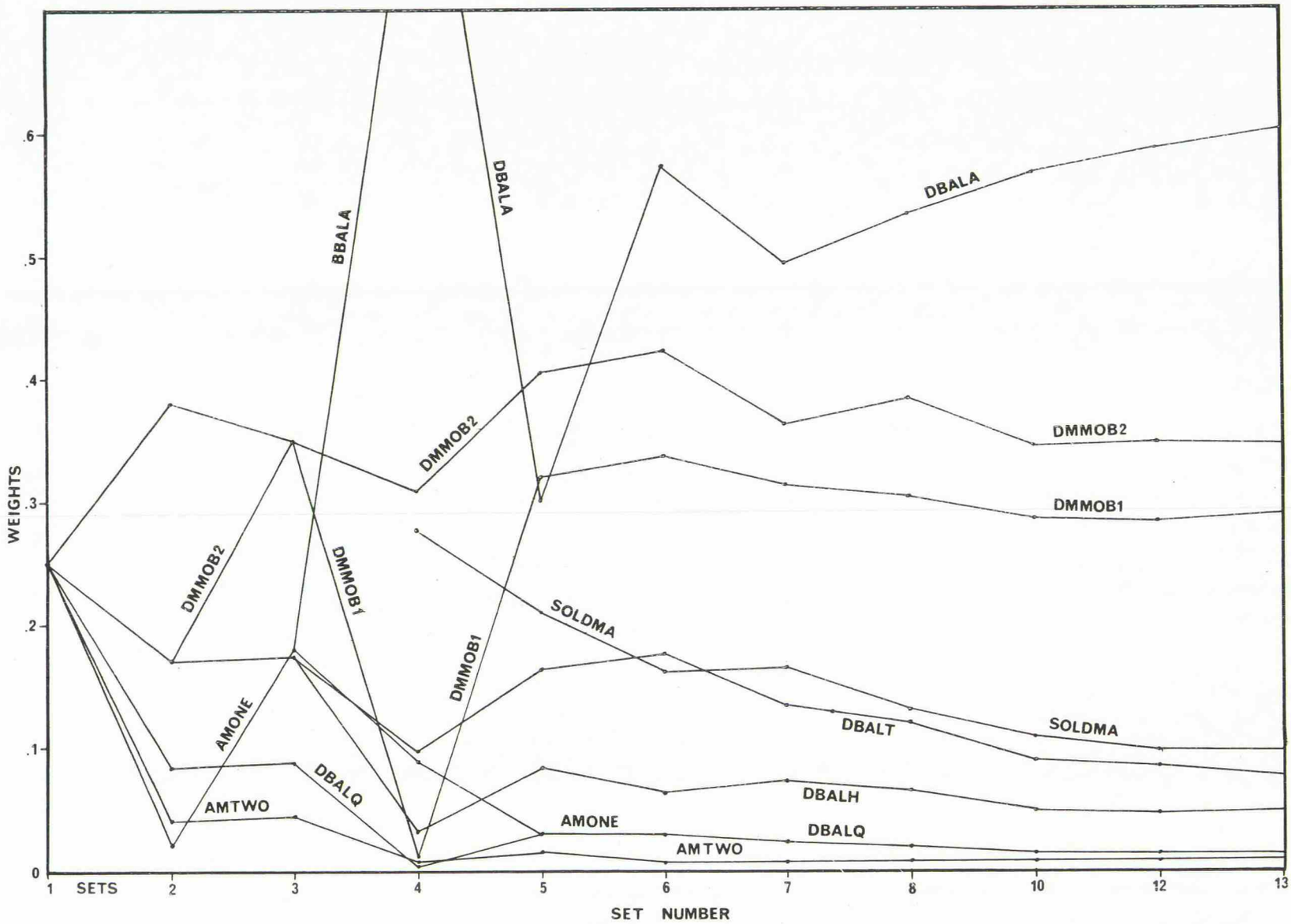


Figure 5.3 Normalized Feature Weights

Training Cycle and Attack Frequency Between Sets

Set	Cycles	Attacks
1	-	-
2	5	2
3	11	5
4	19	9
5	58	18
6	73	34
7	86	40
8	98	46
9	110	52
10	115	55
11	125	61
12	135	67
13	146	73

Table 5.2

At the set 2 point, the logical operators described in section 4.1.4 were added to the learning process to compensate for inconsistencies such as weights on features relating to targets and corridors under attack being less than weights on corresponding features relating to targets and corridors not under attack. Examples in set 2 are TLUND vs TLUNDA and TLONE vs TLONEA.

Selection of key allocations by the TS was maintained through set 3, but the inefficiency and instability previously encountered persisted, so thereafter the key allocations were specified with the attack script. To assure control of decision situations, the key allocation is imposed on the simulation routine regardless of the final selection by the decision procedure and a probability of kill of one (1) is assumed for DM and TDM, but only in the simulation routine, not the decision routine.

Also at the set 3 point, DBALA was added to offset the imbalance between expected loss, measured primarily by AMZERO, and area defense balance. By set 4, it was apparent that key allocations aimed at maintaining PAC coverage above a single DM could not be selected by the decision procedure because it was insensitive to deterioration of PAC coverage up to the point where only a single DM remained. The result was false modification of the weights when such a key allocation was encountered as the learning process attempted to compensate. SOLDMA and SOLEDM were added to provide the necessary information and the decision function began to stabilize.

Final changes to the learning process involved the step size. Frequently the decision function will select an allocation with a value

very close to the key allocation but quite different in missile selection and, hence, in individual feature value. In such a circumstance it is desirable to constrain the ensuing modification to reflect the near equality of the two allocations. At set 5, a constraining factor, equal to the difference between the two values divided by the value of the key allocation, was added and the step size set at the minimum of this factor and the factor based on the history of past performance, as described in section 4.1.3, (a) and (b). To avoid being blocked by this constraint from ever attaining key allocations only slightly better than those selected by the decision function, a minimum bound on step size is necessary and was added a set 6.

VI EXPERIMENTS

In order to measure the effectiveness of the model it is necessary to consider the performance of both the learning and decision processes. The learning process must cause the feature weights to converge and the consequent decision function to provide the basis for consistently good performance by the decision process. Figures 5.2 and 5.3 summarize the evolution of the feature weights. Instability is evident up to set 5 while the learning process and decision function were being modified and the experience was low, fifty-eight cycles and eighteen training attacks at that point. The exaggerated change of DBALA and AMZERO between set 3 and 6 is a result of the fruitless attempt to compensate for the corridor coverage "blind spot" noted in section 5.4 which drove the value of DBALA up and forced AMZERO up also (set 3-4) to protect the expected loss criterion. SOLEDM and SOLDMA were added in set 4 and DBALA and AMZERO made parallel returns (set 5) to slightly above their set 3 levels. By set 6, after seventy-three cycles of thirty-four attacks, the experience factor and the completed feature set have stabilized the learning and thereafter only minor adjustments are made, except to AMZERO, which finally settles down at set 12.

A justification of the relative values of the converged weights requires some analysis. The dominance of AMZERO provides for backup coverage on AM, but not at any cost. Recall that AMZERO is the expected value of AM having no defense if the selected defense fails. For a given AM, if the final defense under consideration is a DM, the .75 kill probability for a DM used in the test environment times the expected value of the target, say 10, would contribute 2.5 to AMZERO, resulting

in a contribution of approximately 3.5 to the decision function. This alone would normally force an alternative action. However, if the final defense is a TDM with a kill probability of .95, the contribution to AMZERO is only .5, and .7 to the decision function, easily exceeded by voiding a DM site, say. Note that AMZERO gives the effect of Heuristic H3 with additional flexibility, and this effect was independently constructed by the learning process.

Several features with low valued weights normally are encountered in groups. For example, when a single DM site is voided and uncovers a PAC, the feature values DBALA and CUND are incremented by one (1). If the PAC leads to an active target, then CUNDA would also be incremented by one (1). If, further, the PAC leads to a locally undefended target, then CTUND would be incremented and CTUNDA, also, if the target was active. The cumulative effect then of voiding a single DM site could be equal to 1.1, and even higher if DMMOB1, DMMOB2, SOLDMA or SOLEDM were effected. Similar grouping occurs when voiding a local defense site, with contributions possible from AMZERO, TLUND, TLUNDA, CTUND and CTUNA. A final observation is that in the environment established to test the model most targets were given a value of 10 so that, with the DM probability of kill set at .75, incremental contributions to AMONE and AMTWO are usually 2.5. Hence, during the early phase of an attack when features with large weights are commonly zero, the contributions of AMONE and AMTWO sway the allocations in favor of expected loss as opposed to balanced defense, measured at this point by DBALQ; however, as the DM sites reach the half depletion point, then DBALH dominates both AMONE and AMTWO.

Clearly, the values of the feature weights are dependent on the defense system environment and in particular on the assigned target values and assumed probabilities of destruction for DM, TDM, and AM.

To test the performance of the decision function, two comparison attacks were run against the model as learning progressed. A summary of the results is given in Tables 6.1, 6.2, and 6.3. Comparison Attack Number 1 consists of fifty-six AM attacking seventeen targets located along the right border of the defense system. All attacks are through plausible attack corridors. Comparison Attack Number 2 is more general, consisting of ninety-nine AM against the thirty-three S, D and M type targets in the system. Attacks are not necessarily through the plausible attack corridors. In both attack scripts, the number of DM and TDM are strictly limited to present a difficult defense problem.

Two techniques were used in simulating the results of AM engagement, one using the probability of destruction specified in the environment for DM and TDM, namely .75 and .95, and one using a probability of destruction of one (1). The second was adopted to provide a better measure of the growth of the decision process by avoiding the "wrong result from the right decision".

The "mesa" phenomenon is observed in Comparison Attack Number 1, Table 6.1. From weight sets 4 through 6 (54 cycles) the same targets are struck, then in set 7 target S1 is saved and not struck again throughout the remainder of the sets. Recall that sets 4 through 6 covered the large fluctuations in DBALA and AMZERO, indicating the mesa was very large since the same overall problem solving ability persisted

Comparison Attack Number 1 with Probability of Kill of 1

Set Number	Final Value	Value Loss	% of Maximum	Hits	Targets Hit
WOE*	526	60	37.5	8	P3, P4, P6, S1, S4, S5
1	546	40	25	5	P3, P4, S1, S4
2	536	50	31.25	6	P3, P4, S1, S4, S5
3	546	40	25	5	P3, P4, S1, S4
4	556	30	18.75	4	P3, P4, S1
5	556	30	18.75	3	P3, P4, S1
6	556	30	18.75	3	P3, P4, S1
7	566	20	12.5	2	P3, P4
8		(identical to 7)			
9	566	20	12.5	2	P3, P4
10		(identical to 9)			
11		"			
12		"			
13		"			

Initial Target Value: 586

Minimum Target Value: 426

Maximum Possible Loss: 160

Heuristics: 1, 2, 4, 5, 9

*Whites of Their Eyes Technique. (see section 1.5)

Table 6.1

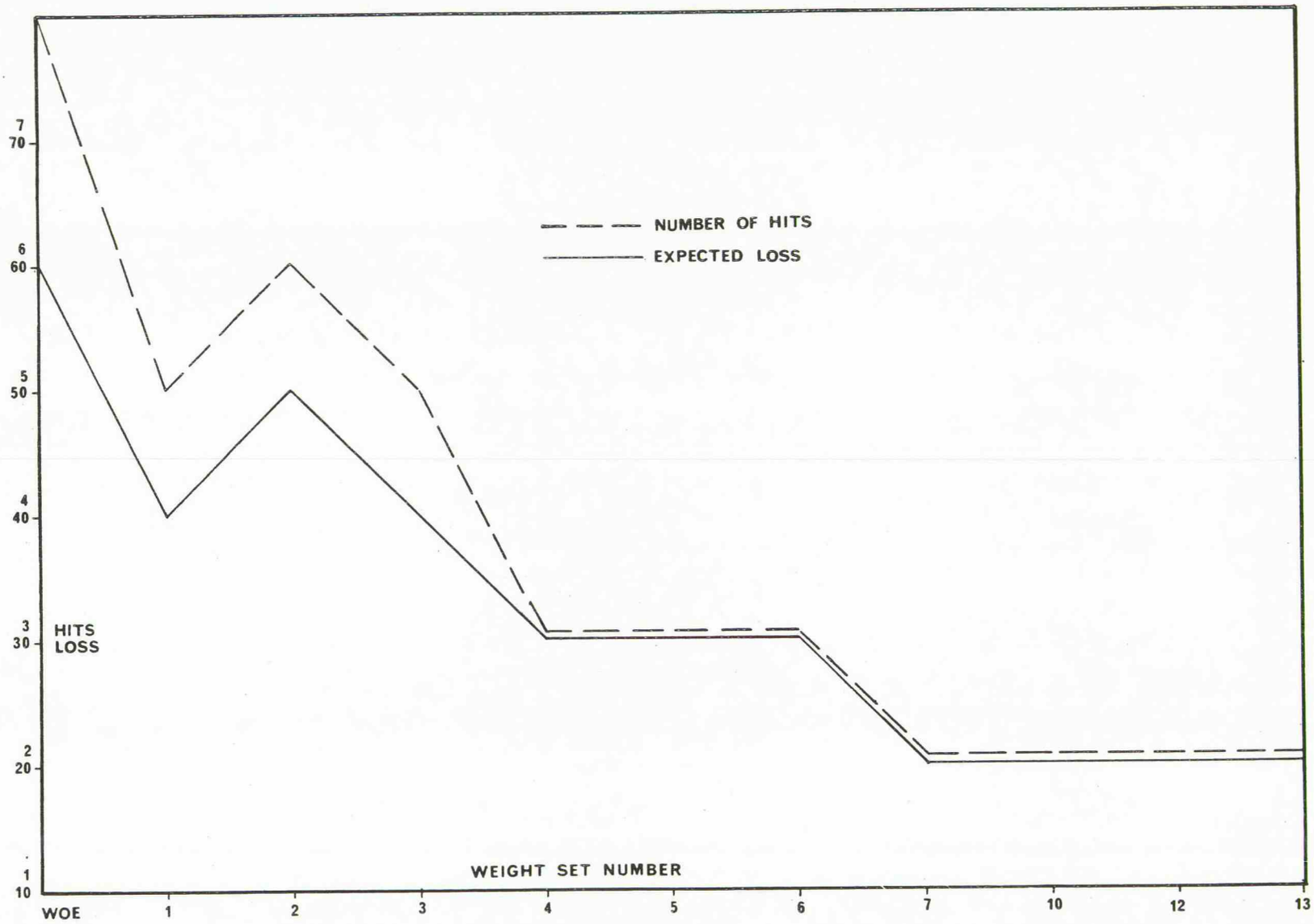


Figure 6.1 Comparison Attack Number 1 with Probability of Kill of 1.

Comparison Attack Number 1

with Probability of Kill of .75 (DM) and .95 (TDM)

Set	Final Value	Value Loss	% of Maximum	Hits	Misses*	Targets Hit
WOE	506	80	50	14	11	P3, P4, P6, PI2, S1, S4, S5, S6
WOE	506	80	50	14	11	(same)
1	506	80	50	17	14	(same)
1	511	75	46.9	11	12	P3, P4, P6, P8, S1, S4, S5, I2
4	526	60	37.5	9	9	P3, P3, P4, P6, S1, S4
4	531	55	34.4	12	15	P3, P4, P6, S1, S4, I2
8	516	70	43.75	10	10	M1, P3, P4, P6, S1, S4, S5, I2
10	516	70	43.75	10	12	P3, P4, P6, S1, S3, S4, S7
12	536	50	31.25	7	10	P2, P3, P4, S2, S6
12	536	50	31.25	8	11	(same)
13	536	50	31.25	8	11	(same)

Heuristics: 1, 2, 5, 9

*The number of times a DM or TDM missed an AM

Table 6.2

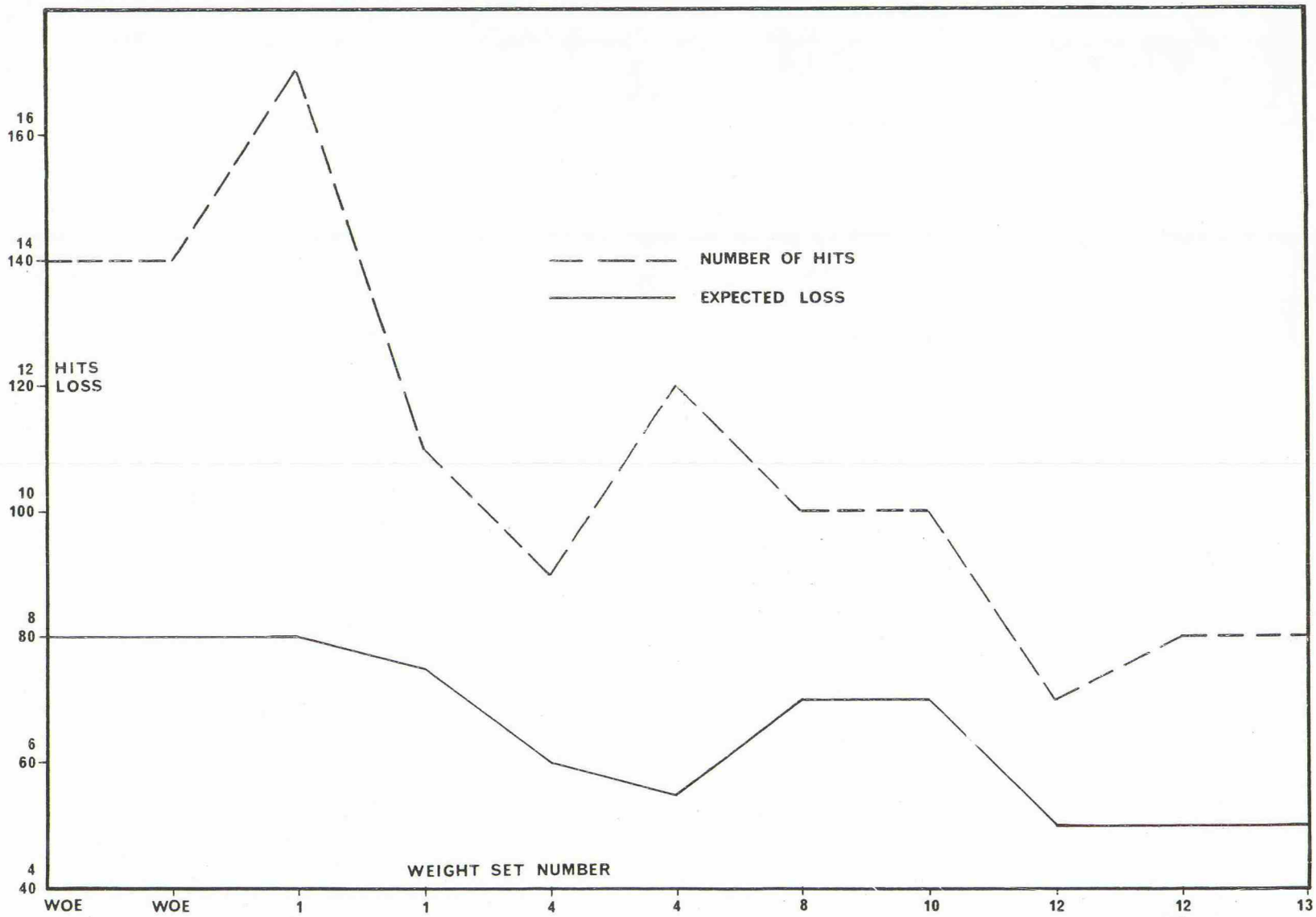


Figure 6.2 Comparison Attack Number 1 with Probability of Kill of .75 (DM) and .95 (TDM)

Comparison Attack Number 2 with Probability of Kill of 1

Set Number	Final Value	Value Loss	% of Maximum	Hits	Targets Hit
WOE	546	50	19.6	7	S2, S3, S5, S9, M11, M12
1	571	25	9.8	3	S2, S7, M11
13	586	10	3.9	1	S2

Initial Target Value: 596

Minimum Target Value: 341

Maximum Possible Loss: 255

Heuristics: 2, 4, 5, 9

Table 6.3

even though the periodic decisions varied widely. Recall, also, that the weights had nearly stabilized by set 7, so uniform performance is to be expected.

Loss of only P3 and P_4 is essentially the optimal solution to Comparison Attack Number 1. P_4 is saturated in the sense that it is attacked by nine AM and has a total of only eight defense opportunities, so is indefensible. P3 is attacked by the final AM, designed to take advantage of an expected void at the sole defending site. The void occurs in period 9 and the AM is launched in period 10. No apparent alteration of the decision function would avoid this situation.

Although there is fluctuation of performance shown on the chance engagement runs, the results of the test runs demonstrate the steady improvement in the decision function throughout the training phase until, at convergence, a near optimal decision process for defense missile allocation in the test environment is attained. Since the test environment was arbitrarily determined, an immediate extension of these results can be made to any other similar defense system environment in the sense that the learning procedure applied in that test will effectively develop an equally good decision function (perhaps with different weights) for that environment.

VII CONCLUDING COMMENTS

Although the results of the experiment described in the preceeding chapters are encouraging, it must be noted that to constrain the scope of the research it was necessary to impose several limiting assumptions on the environmental model and to deliberately omit some promising ideas in the decision process. Some of these deserve comment.

In the environmental model targets were geographically represented as rectangles and damage from successful attacking missiles were accorded only to the target within which the missile landed. Damage assessment programs exist which calculate the blast and fallout effect of nuclear detonations on a target system as a function of weapon yield (megatons), impact point, height at burst and vulnerability of individual targets. Large data files have been created and are being continuously maintained for use by the damage assessment programs, and others, so it is reasonable to assume that an operational anti-ballistic missile system would utilize both the data base and the damage assessment programs to provide expected loss.

Another simplification in the model is that of assigning a single probability of kill to each type of defense missile. The reliability of a defense missile is probably a function of distance to the intercept point, height of AM, location of the AM with relation to the DM site (i.e., is it approaching, going away, overhead, passing abeam of, etc.), configuration of the DM warhead, the booster and the launch pad employed. It is to be expected that reliability information will have been determined from a small sample size due to the expense of testing and the evolution of the missile itself. During an attack the results of every engagement

should be analyzed to provide continual updating of reliability for use by the decision function.

Within the decision process little emphasis was placed on the selection of a terminal defense missile either in comparing alternative allocations or engaging. Since TDM sites defending a target area are highly interdependent because of overlapping target coverage, a decision function is needed to select which one to use against AM attacking targets defended by more than one site. Determining the decision function is a non-trivial problem, perhaps of the same scope as the one being reported, and provides an interesting and important area for additional research.

When considering future events, the decision process assumes the enemy has an infinite number of AM by always scanning every plausible attack corridor. However, plausible attack corridors lead from geographical locations where it is known that the enemy has ballistic missiles and estimates of the number of missiles can be made from intelligence information. An obvious improvement in the decision process is to count AM originating at each of the locations and those assessed destroyed by friendly forces and eliminate plausible attack corridors as the missile source becomes depleted. For example, the missile ships and submarines which menace the coasts carry limited, predictable numbers of missiles. After they have launched all of their missiles, the plausible attack corridors established to reflect their capability need no longer be considered. Not only would the decision process be improved by having better information, but the search time would also be reduced.

A final comment concerns a posteriori learning; learning while an attack is in progress. Because of the nature of nuclear war and the unlikelihood that a nation will gain practical experience in defending itself from a missile attack, the training of the decision function will be done by using models and simulation as was done above. Once an attack is initiated, the performance of the decision function will be critically dependent on how well the training model reflected that particular attack, unless the decision process can adjust itself under fire. The mechanics for making the adjustments are available in the learning procedure and can be applied in an off-line mode by reviewing past decisions in the light of all currently available information, adjusting the decision function weights as in Chapter IV. To be responsive, key allocations should be selected automatically (see section 4.1.1 for a discussion of this technique), since time will probably not permit human analysis to specify key allocations. The effectiveness of a posteriori learning is another subject for future research.

BIBLIOGRAPHY

[0] Clarkson, G. P. E., 1962, Portfolio Selection: A Simulation of Trust Investment, Englewood Cliffs, N. J.: Prentice-Hall. Also condensed in [1] pp. 347-371.

[1] Feigenbaum, E. A. and Feldman, J. (Ed.), 1963, Computers and Thought, McGraw-Hill.

[2] Feigenbaum, E. A., 1961, The Simulation of Verbal Learning Behavior, Proceedings of the Western Joint Computer Conference, pp. 121-132.

[3] Gyr, J. W., 1960, An Investigation into, and Speculation about, the Formal Nature of a Problem Solving Process, Behavioral Science, January, pp. 39-59.

[4] Karge, R. L. and Thompson, G. L., 1964, A Heuristic Approach to the Travelling Salesman Problem, Management Science 10, p. 225.

[5] Kuehn, A. A. and Hamburger, M. J., 1963, A Heuristic Program for Locating Warehouses, Management Science 9, p. 643.

[6] Minsky, M. L., 1961, Steps Toward Artificial Intelligence, Proceedings of the IRE, Special Computer Issues. Also in [1] pp. 406-450.

[7] Newell, A., Shaw, J. C. and Simon, H. A., 1959, Report on a General Problem-Solving Program, Proceedings of the International Conference on Information Processing, UNESCO House, pp. 256-264.

[8] Newell, A. and Simon, H. A., 1961, GPS, a Program that Simulates Human Thought, Lernende Automaten, Munich: R. Oldenbourg KG. Also in [1] pp. 279-293.

[9] _____ and _____, 1958, Heuristic Problem Solving: The Next Advance in Operations Research, Journal of the Operations Research Society of America 6, pp. 1-10.

[10] Samuel, A. L., 1959, Some Studies in Machine Learning Using the Game of Checkers, IBM Journal of Research and Development 3, pp. 210-219. Also in [1] pp. 71-105.

[11] Schwartz, E. S., 1964, A Heuristic Procedure for Parallel Sequencing with Choice of Machines, Management Science 10, p. 767.

[12] Selfridge, O. G., 1959, Pandemonium: A Paradigm for Learning, Proceedings of the Symposium on Mechanization of Thought Processes, pp. 511-529.

[13] Selfridge, O. G. and Neisser, U., 1960, Pattern Recognition by Machine, Scientific American 203, pp. 60-68. Also in [1] pp. 237-250.

[14] Tonge, F. M., 1963, Balancing Assembly Lines Using the General Problem Solver, Symposium on Simulation Models: Methodology and Applications to the Behavioral Sciences, pp. 139-151.

[15] Tonge, F. M., 1961, A Heuristic Program for Assembly Line Balancing, Prentice-Hall.

[16] Uhr, L. and Vossler, C., 1961, A Pattern Recognition Program that Generates, Evaluates and Adjusts its Own Operators, Annals of the New York Academy of Science, pp. 555-569. Also in [1] pp. 251-258.

APPENDIX A

Program Listing

```
          BEGIN                                                                 00000010        ARC00010
      COMMENT OUTER BLOCK OF ACTS;                                              00000020        ARC00010
INTEGER       MAMDATA,NAMDATA,MSQRDTA,NSQRDTA ,MATTACK,NATTACK,NDMSPEC,         00000030        ARC00020
              MTGT, NTDLIST,MDMLOCO,NTYPE,MPAMDTA,NPAMDTA,MPACDTA,              00000040        ARC00030
              NFEATURES;                                                        00000050        ARC00030
REAL ARRAY    XXXXXX[0:1];                                                      00000060        ARC00040
DEFINE        TYPATK=XXXXXX[0]#;                                                00000070        ARC00040
FILE          DATAIN "SRMCARC"(2,10),DATAOUT 4(2,15);                           00000080        ARC00050
FILE          PACDTAF DISK SERIAL(1:15] "CALLERO" "PACDTA" (1,30);              00000090        ARC00060
FILE          PACDMSF DISK SERIAL(1:8] "CALLERO" "PACDMS" (1,30);               00000100        ARC00070
FILE OUT      PUNCHF O(2,10);                                                   00000110
LABEL         STARTNEWPROBLEM,ENDOFRUN;                                         00000120
STARTNEWPROBLEM:WRITE(DATAOUT[PAGE1);                                           00000130
              READ(DATAIN,/,MAMDATA,NAMDATA,MSQRDTA,NSQRDTA,MATTACK,            00000140        ARC00080
              NATTACK,NDMSPEC, MTGT,NTDLIST,MDMLOCO,NTYPE,MPAMDTA,              00000150        ARC00090
              NPAMDTA,MPACDTA,NFEATURES);                                       00000160        ARC00090
          BEGIN                                                                 00000170        ARC00090
      COMMENT MAIN BODY OF ACTS;                                                00000180        ARC00100
INTEGER       I,J,K,KA,NTGT,N3, NTYPT,NPACDTA,NPAC,AMDET,AMNOW,                 00000190        ARC00110
              ESTOTAM, LASTPROW,LASTPCOL,LASTALT, TWOCOR,THREECOR,KB,           00000200        ARC00120
              KC,KD,KE,KF,KG, ALTERSONE,ITDMZERO,ITDMONE,ITDMZEROA,             00000210        ARC00130
              ITDMONEA, MINATT,CURATT, REPEAT,TREPEAT, AMIND,CYCLE,             00000220        ARC00140
              INITUNC, V1,V3,TV1,TV3,LV1,LV3, WRAC,NPACDMS,                     00000230        ARC00150
              ACTIVEINDEX, FUTALT,NEXTALT, FINALT, MAXAMDT, NUMGRPS,            00000240        ARC00160
              GRPNOW,GRPCUTOFF,SAVECYCLE, NDM,MSQR,NSQR,PERIOD,NAM,             00000250        ARC00170
              ALTX,L,TGTAFF,TDMAFF,EVALCNT,SUBEVALCNT;                          00000260        ARC00170
BOOLEAN       HEURISTIC2,HEURISTIC3,HEURISTIC4,FEAS,TDMONE,TDMZERO,             00000270        ARC00180
              ACTVTGT, SOOK, LEARNING,FLAG, AMFLAG,PACIN, REBUILDPAC,           00000280        ARC00190
              FLGA,HEURISTIC8, BOOK1,BOOK2,NEWWATE,ADJWATE,COMPARUN,            00000290        ARC00200
              OUT1,OUT2,OUT3, OUT7,HEURISTIC9,HEURISTIC1, HEURISTIC5,           00000300        ARC00220
              HEURISTIC6,HEURISTIC10,ANSWER,ANSCOMB,ANSWAS,LSAVE,               00000310        ARC00220
              HEURISTIC7,NODEFENSE,NOTGT,TOOLATE,FIRSTPASS,RSEED,BPAC;          00000320        ARC00230
REAL          V2,TV2,LV2,CW,CWO,MAXPERIOD,MINTGTVAL,TOTVALUE,FINVALUE,          00000330        ARC00240
              TC,TD, T,TA,TB,TME,TDMPROB,DESTPROB,VALU,XVALU,                   00000340        ARC00250
              RANDOMSEED;                                                       00000350        ARC00250
INTEGER ARRAY SQRX,SQRY[0:100],INTC[0:MTGT ],TDLIST[0:NTDLIST],                 00000360        ARC00260
              ANSINDEX[0:10],REPWATE[0:NFEATURES],BOOKANSWER[0:30,0:10]         00000370        ARC00270
              , DMVOID[0:NTYPE],DMCOUNT[0:MDMLOCO],ALTERS[0:MAMDATA,0:          00000380        ARC00280
              11], SDLOCUL[0:MDMLOCO×NTYPE], SQRTGT[0:MSQRDTA,0:NSQRDTA,        00000390        ARC00290
              0:15], BESTCOMB,LBESTCOMB[0:30],ACTGT[0:2,0:7],GROUPS[0:          00000400        ARC00300
              MDMLOCO], RUNID[0:4], DLIST[0:NTDLIST],TDMTGT[0:MTGT,0:           00000410        ARC00310
              15], BESTALT[0:MAMDATA], PACDMS[0:239],SDMV[0:NTYPE],             00000420        ARC00320
              SOLEDM,SOLDMA[0:1], TGTIAD[0:MTGT,0:5],SQRDTA[0:MSQRDTA,          00000430        ARC00330
              0:NSQRDTA],AMDATA[0:MAMDATA, 0:NAMDATA],PACDTA[0:MPACDTA,         00000440        ARC00340
              0:63],PAMDTA[0:MPAMDTA,0:NPAMDTA];                                00000450        ARC00340
REAL ARRAY    INTA,INTB[0:MTGT ],ATTACK[0:MATTACK,0:NATTACK], PFEAT,            00000460        ARC00360
              PFEATURE ,LFEATURE,LPFEATURE,INPFEAT[0:NFEATURES],                00000470        ARC00360
              SAVEINF[0:150], DMLOCO[0:MDMLOCO,0:NTYPE+3],TGTMNX[0:             00000480        ARC00370
              MTGT],TGTLCV[0:MTGT,0:4], WATE,XWATE,INFEAT,INLFEAT[0:            00000490        ARC00380
              NFEATURES], DMSPEC[0:2×NTYPE],FEATURE,TFEAT,XFEAT[0:              00000500        ARC00390
              NFEATURES];                                                       00000510        ARC00390
ALPHA         CODE;                                                             00000520        ARC00400
LABEL         FULL,FINISH,SIMRESULT,NEXTPERIOD,REPORTS,PROCESS;                 00000530        ARC00410
FORMAT        TGTIN(A4,A6,X3,A6,X3,4 I3,X2,4 R7.0,X2,R4.0),A(A3),               00000540        ARC00420
              LBEST("LBESTCOMB",X4,15(I2,I2,X3)), BESTC("BESTCOMB ",X4,         00000550        ARC00440
              15(I2,I2,X3)), BESTA("BESTALT  ",X4,15(I2,I2,X3)),                00000560        ARC00450
              NOHEUR("HEURISTIC",I2," IS SPECIFIED BUT IS ALWAYS SET")          00000570        ARC00460
              , KILLAM(/"AM ",I3," DESTROYED DURING PERIOD",I4," BY ",          00000580        ARC00470
              A3," SITE",I4), COMBINC                                           00000590        ARC00480
              "TOTAL ALTERNATIVES WITH HEURISTICS",I10,",  WITHOUT",            00000600        ARC00480
              I10/), FUTDEST(/"AT PERIOD",I4,                                   00000610        ARC00490
```

```
                " THERE IS NO DEFENSE AGAINST AM",I4," ATTACKING TARGET "00000620        ARC00500
                ,A6,".  THE RESULT WILL BE AS FOLLOWS "), DESTTGT(        00000630        ARC00510
                "TARGET ",A6," STRUCK DURING PERIOD",I4," BY AM",I4,      00000640        ARC00510
                ".  TARGET VALUE IS NOW",F11.7/);                         00000650        ARC00520
SWITCH FORMAT  HEUR← ("DUMMY"),("HEURISTIC 1  USE PAC LOOKAHEAD"), (      00000660        ARC00522
       "HEURISTIC 2  USE TDM ON AM ARRIVING AT TARGET IN CURRENT PERIOD"00000670        ARC00522
                ), ("HEURISTIC 3  ALWAYS PROVIDE DOUBLE DEPTH DEFENSE C",00000680        ARC00523
                "OVERAGE"), (                                             00000690        ARC00524
                "HEURISTIC 4  USE KILL PROBABILITY OF 1 FOR DM AND TDM") 00000700        ARC00524
                , ("HEURISTIC 5  APPLY INDEPENDENT AM PLANNING"), (       00000710        ARC00526
                "HEURISTIC 6  APPLY PRESPECIFIED DM GROUP PLANNING"), (   00000720        ARC00527
                "HEURISTIC 7  USE WHITES OF THEIR EYES DEFENSE STRATEGY")00000730        ARC00527
                , ("HEURISTIC 8  DEFEND ONLY TARGETS WITH EXPECTED VALU",00000740        ARC00528
                "E > MINTGTVAL"), (                                       00000750        ARC00529
                "HEURISTIC 9  SELECT ON DECISION FUNCTION VALUE ONLY");   00000760        ARC00529
DEFINE          FORI=FOR I←0 STEP 1 UNTIL #,                              00000770        ARC00530
                FORI1=FOR I←1 STEP 1 UNTIL #,                             00000780        ARC00530
                FORJ=FOR J←0 STEP 1 UNTIL #,                              00000790        ARC00540
                FORJ1=FOR J←1 STEP 1 UNTIL #,                             00000800        ARC00540
                FORK=FOR K←0 STEP 1 UNTIL#,                               00000810        ARC00550
                PACDTAN3=PACDTA[N3,[36:6],N3,[42:6]]#;                    00000820        ARC00550
                                                                          00000830        ARC00551
COMMENT CHEKBIT IS TRUE IF BIT B IN WORD A IS SET TO 1;                   00000840        ARC00551
BOOLEAN STREAM PROCEDURE CHEKBIT(A,B);                                    00000850        ARC00560
VALUE           B;                                                        00000860        ARC00560
                 BEGIN                                                    00000870        ARC00560
                   SI←A; SKIP B SB; TALLY←0;                              00000880        ARC00570
                   IF SB THEN TALLY←1; CHEKBIT←TALLY;                     00000890        ARC00570
                 END CHEKBIT;                                             00000900        ARC00570
                                                                          00000910        ARC00571
COMMENT SETBIT SETS BIT B IN WORD A TO 1;                                 00000920        ARC00571
STREAM PROCEDURE SETBIT(A,B);                                             00000930        ARC00580
VALUE           B;                                                        00000940        ARC00580
                 BEGIN                                                    00000950        ARC00580
                   DI←A; SKIP B DB; DS←SET;                               00000960        ARC00590
                 END SETBIT;                                              00000970        ARC00590
                                                                          00000980        ARC00591
COMMENT CLEARBIT SETS BIT B IN WORD A TO 0;                               00000990        ARC00591
STREAM PROCEDURE CLEARBIT(A,B);                                           00001000        ARC00600
VALUE           B;                                                        00001010        ARC00600
                 BEGIN                                                    00001020        ARC00600
                   DI←A; SKIP B DB; DS←RESET;                             00001030        ARC00610
                 END CLEARBIT;                                            00001040        ARC00610
STREAM PROCEDURE PICKCHARS(SRCE,DEST,SCHR,DCHR,NCHR);                     00001050        ARC00620
VALUE           SCHR,DCHR,NCHR;                                           00001060        ARC00630
                 BEGIN                                                    00001070        ARC00630
      COMMENT TRANSFER NCHR CHARACTERS BEGINNING WITH THE SCHR CHARACTER  00001080        ARC00640
 IN SRCE TO DEST BEGINNING AT THE DCHR CHARACTER IN DEST;                 00001090        ARC00650
                   SI←SRCE; SI←SI+SCHR; SI←SI-1; DI←DEST;                 00001100        ARC00660
                   DI←DI+DCHR; DI←DI-1; DS←NCHR CHR;                      00001110        ARC00660
                 END PICKCHARS;                                           00001120        ARC00670
                                                                          00001130        ARC00671
COMMENT BREAKDMWORD UNPACKS DMWORD IN THE FORM STORED IN TABLE AMDATA      00001140        ARC00671
        AND PUTS THE NUMBER OF RELEVANT TIME PERIODS IN A, FIRST EFFEC-   00001150        ARC00672
        TIVE PERIOD IN B, DM SITE NO. IN C AND DM TYPE IN D;              00001160        ARC00673
STREAM PROCEDURE BREAKDMWORD(DMWORD,A,B,C,D);                             00001170        ARC00680
                 BEGIN                                                    00001180        ARC00680
                   SI←DMWORD; SI←SI+2; DI←A; DS←8 LIT "0";                00001190        ARC00690
                   DI←B; DS←8 LIT "0"; DI←C; DS←8 LIT "0";                00001200        ARC00690
                   DI←D; DS←8 LIT "0"; DI←A; DI←DI+7; DS←CHR;             00001210        ARC00700
                   DI←B; DI←DI+6; DS←2 CHR; DI←C; DI←DI+6;                00001220        ARC00700
```

```
                  DS←2 CHR; DI←0; DI←DI+7; DS← CHR;                    00001230    ARC00710
                END BREAKDMWORD;                                       00001240    ARC00710
                                                                       00001250    ARC00721
COMMENT ACTIVE IS TRUE IF THE TARGET TYPE GIVEN IN A INCLUDES A CHARACTR00001260    ARC00721
        IN THE ACTIVE TARGET LIST GIVEN IN B;                          00001270    ARC00722
BOOLEAN STREAM PROCEDURE ACTIVE(A,B);                                  00001280    ARC00720
                BEGIN                                                  00001290    ARC00720
                  TALLY←0; SI←A; B(IF SC="0" THEN JUMP OUT;            00001300    ARC00730
                  DI←B; B(IF SC=DC THEN TALLY←1;                       00001310    ARC00730
                  IF TOGGLE THEN JUMP OUT; SI←SI-1);                   00001320    ARC00730
                  IF TOGGLE THEN JUMP OUT; SI←SI+1);                   00001330    ARC00740
                  ACTIVE←TALLY;                                        00001340    ARC00740
                END ACTIVE;                                            00001350    ARC00740
BOOLEAN STREAM PROCEDURE ISIN(B,C);                                    00001360    ARC00750
                BEGIN                                                  00001370    ARC00750
                                        COMMENT TRUE IF CHAR C IS IN B;00001380    ARC00750
                  SI←B; DI←C; DI←DI+7; TALLY←0;                        00001390    ARC00760
                  B(IF SC=DC THEN TALLY←1;                             00001400    ARC00760
                  IF TOGGLE THEN JUMP OUT; DI←DI-1);                   00001410    ARC00770
                  ISIN←TALLY;                                          00001420    ARC00770
                END ISIN;                                              00001430    ARC00770
INTEGER STREAM PROCEDURE OCCUR(VECTOR,N,WD,VC,WC,NC);                  00001440    ARC00780
VALUE           VC,WC,NC,N;                                            00001450    ARC00780
                BEGIN                                                  00001460    ARC00780
      COMMENT OCCUR IS INDX IF THE NC CHARACTERS IN WD STARTING AT     00001470    ARC00790
 CHAR WC OCCUR IN THE FIRST N WORDS OF VECTOR IN LOCATION VC,IF THERE   00001480    ARC00800
 IS NO MATCH THEN OCCUR=N;                                             00001490    ARC00810
                  SI←WD; SI←SI+WC; SI←SI-1; DI←VECTOR;                 00001500    ARC00820
                  DI←DI+VC; DI←DI-1; TALLY←0;                          00001510    ARC00820
                  NC(IF NC SC=DC THEN JUMP OUT; TALLY←TALLY+1;         00001520    ARC00830
                  SI←SI-NC; DI←DI-NC; DI←DI+8); OCCUR←TALLY;           00001530    ARC00840
                END OCCUR;                                             00001540    ARC00840
                                                                       00001550    ARC00841
COMMENT PACKWORD PACKS THE LAST CHARACTER IN THE FIRST 8 WORDS IN VECTOR00001560    ARC00841
        INTO WORD,UNPACK REVERSES THE PROCESS;                         00001570    ARC00842
STREAM PROCEDURE PACKWORD(WORD,VECTOR);                                00001580    ARC00850
                BEGIN                                                  00001590    ARC00850
                  DI←WORD; SI←VECTOR; B(SI←SI+7; DS←CHR);              00001600    ARC00860
                END PACKWORD;                                          00001610    ARC00870
STREAM PROCEDURE UNPACK(WD,V);                                         00001620    ARC00880
                BEGIN                                                  00001630    ARC00880
                  SI←WD; DI←V; B(DS←7 LIT "0000000";                   00001640    ARC00880
                  DS←CHR)                                              00001650    ARC00890
                END UNPACK;                                            00001660    ARC00890
                                                                       00001670    ARC00891
COMMENT MOVEWORD STORES WORD A IN WORD B;                              00001680    ARC00891
STREAM PROCEDURE MOVEWORD(A,B);                                        00001690    ARC00900
                BEGIN                                                  00001700    ARC00900
                  SI←A; DI←B; DS←WDS;                                  00001710    ARC00900
                END MOVEWORD;                                          00001720    ARC00900
                                                                       00001730    ARC00901
COMMENT SETONES SETS EACH BIT IN WORD W TO 1;                          00001740    ARC00901
STREAM PROCEDURE SETONES(W);                                           00001750    ARC00910
                BEGIN                                                  00001760    ARC00910
                  DI←W; DS←48 SET;                                     00001770    ARC00910
                END SETONES;                                           00001780    ARC00910
                                                                       00001790    ARC00911
COMMENT PACVALUE IS A DUMMY PROCEDURE TO PERMIT FUTURE EXPANSION;      00001800    ARC00911
REAL PROCEDURE PACVALUE;                                               00001810    ARC00930
                BEGIN                                                  00001820    ARC00930
                  PACVALUE←0;                                          00001830    ARC00930
```

```
                END PACVALIS;                                                    00001840    ARC00930
                                                                                 00001850    ARC00931
COMMENT RANDOM IS A REAL VALUED PSEUDO RANDOM NO. BETWEEN 0 AND 1;               00001860    ARC00931
REAL PROCEDURE RANDOM;                                                           00001870    ARC00940
                BEGIN                                                            00001880    ARC00940
STREAM PROCEDURE  SETOR(R);                                                      00001890    ARC00950
                  BEGIN                                                          00001900    ARC00950
                    DI←R; 3 CDS + 2 RESET; DS + 1 SET; ;                         00001910    ARC00960
                    DS + 39 RESET ;                                              00001920    ARC00970
                  END SETOR;                                                     00001930    ARC00970
REAL              A;                                                           00001940    ARC00980
                  SETOR(A);                                                      00001950    ARC00980
                  RANDOM←REAL((XSEED←BOOLEAN(2045*REAL(XSEED)+                   00001960    ARC00990
                  211527139) AND BOOLEAN(134217727)) OR BOOLEAN(A))              00001970    ARC01000
                END RANDOM;                                                      00001980    ARC01000
                                                                                 00001990    ARC01001
COMMENT COVERED IS TRUE IF TARGET SPECIFIED BY I IS LOCALLY DEFENDED;            00002000    ARC01001
BOOLEAN PROCEDURE COVERED(I);                                                    00002010    ARC01010
VALUE             I;                                                             00002020    ARC01010
INTEGER           I;                                                             00002030    ARC01010
                  BEGIN                                                          00002040    ARC01010
INTEGER             J;                                                           00002050    ARC01020
LABEL               FIN;                                                         00002060    ARC01020
                    FOR J←2 STEP 1 UNTIL 5 DO IF TDLIST[TGTTAD[I,J]]≠0           00002070    ARC01030
                    THEN                                                         00002080    ARC01030
                      BEGIN                                                      00002090    ARC01030
                        COVERED← TRUE; GO TO FIN;                                00002100    ARC01040
                      END;                                                       00002110    ARC01040
                    COVERED←FALSE;                                               00002120    ARC01040
FIN:                END COVERED;                                                 00002130    ARC01040
                                                                                 00002140    ARC01051
COMMENT REMDM IS THE NUMBER OF DEFENSE OPPORTUNITIES AFTER PERIOD PER            00002150    ARC01051
        FOR AM WITH DATA STORED IN ROW ROW OF TABLE ALTERS;                      00002160    ARC01052
INTEGER PROCEDURE REMDM(ROW,PER);                                                00002170    ARC01050
VALUE             ROW,PER;                                                       00002180    ARC01050
INTEGER           ROW,PER;                                                       00002190    ARC01050
                  BEGIN                                                          00002200    ARC01050
        COMMENT COUNT THE DEFENSE OPPORTUNITIES FOLLOWING PERIOD PER;            00002210    ARC01060
INTEGER             I,K,KA,KB,KC,KD,KE,KF;                                       00002220    ARC01070
LABEL               FIN, L2,L1;                                                  00002230    ARC01070
                    KF←ALTERS[ROW,3];                                            00002240    ARC01070
                    IF PER=AMDATA[ALTERS[ROW,0],2] THEN                          00002250    ARC01080
                      BEGIN                                                      00002260    ARC01080
                        K←0; GO TO FIN                                           00002270    ARC01080
                      END;                                                       00002280    ARC01080
                    KA←KB←KC←KD←K←0;                                             00002290    ARC01090
                    KE←AMDATA[ALTERS[ROW,0],3];                                  00002300    ARC01090
                    FOR I←2,3,4,5 DO IF TDLIST[TGTTAD[KF,I]]>0 THEN              00002310    ARC01100
                      BEGIN                                                      00002320    ARC01100
                        K←1; GO TO L1                                            00002330    ARC01100
                      END;                                                       00002340    ARC01100
L1:                   FOR I←5 STEP 1 UNTIL KF DO                                 00002350    ARC01110
                      BEGIN                                                      00002360    ARC01110
                        BREAKDMWORD(ALTERS[ROW,I],KA,KB,KC, KD);                 00002370    ARC01120
                        IF KB=0 THEN GO TO L2; KE←KB+KA;                         00002380    ARC01120
                        IF KE>PER AND NOT CHEKBIT(DMVOID[KD] ,KC) THEN K←K+1;    00002390    ARC01130
                        KE←(IF PER<KB THEN KB-1 ELSE PER);                       00002400    ARC01130
L2:                   END;                                                       00002410    ARC01130
FIN:                REMDM←K;                                                     00002420    ARC01140
                END REMDM;                                                       00002430    ARC01140
                                                                                 00002440    ARC01151
```

```
COMMENT HEUR2 IS TRUE IF HEURISTIC H2 APPLIES TO THE AM STORED IN AMDATA  00002450  ARC01151
        ROW I DURING PERIOD P;                                             00002460  ARC01152
BOOLEAN PROCEDURE HEUR2(I,P);                                              00002470  ARC01150
VALUE           I,P;                                                       00002480  ARC01150
INTEGER         I,P;                                                       00002490  ARC01150
                BEGIN                                                      00002500  ARC01150
                  HEUR2←FALSE; IF NOT HEURISTIC2 THEN P←0;                 00002510  ARC01160
                  IF P=AMDATA[I,2] AND AMDATA[I,4]≥0 THEN                  00002520  ARC01170
                   BEGIN                                                   00002530  ARC01170
                     I←AMDATA[I,3];                                        00002540  ARC01170
                     FOR P←2,3,4,5 DO IF TOLIST[TGTTAD[I,P]]>0 THEN        00002550  ARC01180
                     HEUR2←TRUE                                            00002560  ARC01180
                   END                                                     00002570  ARC01180
                END;                                                       00002580  ARC01180
                                                                           00002590  ARC01181
COMMENT HEUR3 IS THE INDEX IN ALTERS ROW N WHICH SATISFIES THE PROVISION   00002600  ARC01181
        OF HEURISTIC H3;                                                   00002610  ARC01182
INTEGER PROCEDURE HEUR3(N);                                                00002620  ARC01190
VALUE           N;                                                         00002630  ARC01190
INTEGER         N;                                                         00002640  ARC01190
                BEGIN                                                      00002650  ARC01190
      COMMENT HEUR3 ID INDEX OF ALTERS ROW N SATISFYING HEURISTIC 3;       00002660  ARC01200
INTEGER           A,B,C,D,I,J,K,L,AN3;                                     00002670  ARC01210
LABEL             FIN,SETH;                                                00002680  ARC01210
                  AN3←ALTERS[N,3];                                         00002690  ARC01210
                  IF NOT HEURISTIC3 OR AN3=5 THEN                          00002700  ARC01220
                   BEGIN                                                   00002710  ARC01220
                     HEUR3←AN3; GO TO FIN;                                 00002720  ARC01220
                   END;                                                    00002730  ARC01220
                  L←ALTERS[N,1]+1; IF L=1 THEN L←5; K←0;                   00002740  ARC01230
                  IF NOT BPAC THEN                                         00002750  ARC01230
                   BEGIN                                                   00002760  ARC01230
                     FOR J←AN3 STEP -1 UNTIL L DO                          00002770  ARC01240
                      BEGIN                                                00002780  ARC01240
                        BREAKDMWORD(ALTERS[N,J],A,B,C,D);                  00002790  ARC01250
                        IF A≠0 OR K≠0 AND K≠B THEN GO TO SETH;             00002800  ARC01270
                        K←B;                                               00002810  ARC01270
                      END FORJ;                                            00002820  ARC01270
                     IF J=4 THEN                                           00002830  ARC01280
                      BEGIN                                                00002840  ARC01280
                        J←5; GO TO SETH                                    00002850  ARC01280
                      END;                                                 00002860  ARC01280
                     HEUR3←J; GO TO FIN;                                   00002870  ARC01280
                   END ELSE                                                00002880  ARC01280
                   BEGIN                                                   00002890  ARC01280
                     FOR J←AN3 STEP -1 UNTIL 5 DO                          00002900  ARC01290
                      BEGIN                                                00002910  ARC01290
                        BREAKDMWORD(ALTERS[N,J],A,B,C,D);                  00002920  ARC01300
                        IF DMLOCO[C,D+3]>1 THEN GO TO SETH                 00002930  ARC01310
                      END FORJ;                                            00002940  ARC01310
                     HEUR3←AN3; GO TO FIN;                                 00002950  ARC01310
                   END;                                                    00002960  ARC01310
SETH;             K←B;                                                     00002970  ARC01320
                  FOR J←J+1 STEP 1 UNTIL AN3 DO                            00002980  ARC01320
                   BEGIN                                                   00002990  ARC01320
                     BREAKDMWORD(ALTERS[N,J],A,B,C,D);                     00003000  ARC01330
                     IF B≠K THEN                                           00003010  ARC01340
                      BEGIN                                                00003020  ARC01340
                        HEUR3←J-1; GO TO FIN;                              00003030  ARC01340
                      END                                                  00003040  ARC01340
                   END FORJ;                                               00003050  ARC01340
```

```
                HEUR3←AN3;                                                  00003060    ARC01340
FIN:          END HEUR3;                                                    00003070    ARC01350
                                                                            00003080    ARC01371
COMMENT FINDGROUPS ESTABLISHES PLANNING GROUPS FOR AM STORED IN TABLE       00003090    ARC01371
        ALTERS BEGINNING WITH ROW B AND INCLUDING ROW E;                    00003100    ARC01372
PROCEDURE       FINDGROUPS(B,E);                                            00003110    ARC01370
VALUE           B,E;                                                        00003120    ARC01380
INTEGER         B,E;                                                        00003130    ARC01380
                 BEGIN                                                      00003140    ARC01380
       COMMENT SET UP GROUPS ,   NUMGRPS AND ALTERS[X,11];                  00003150    ARC01390
INTEGER           I,J,K,L,P,IG,X,Y,Z,KA,KB;                                 00003160    ARC01400
LABEL             LENDK;                                                    00003170    ARC01400
INTEGER ARRAY     WGRP[0:NDM,0:NDM];                                        00003180    ARC01410
                  FOR K←B STEP 1 UNTIL E DO                                 00003190    ARC01420
                   BEGIN                                                    00003200    ARC01420
                     Z←ALTERS[K,3];                                         00003210    ARC01420
                     IF ALTERS[K,Z]=0 THEN Z←Z-1;                           00003220    ARC01430
                     IF Z<5 THEN GO TO LENDK;                               00003230    ARC01430
                     FOR J←5 STEP 1 UNTIL Z DO BREAKDMWORD(ALTERS[K,J],     00003240    ARC01440
                     I,I,WGRP[0,J-4],I); WGRP[0,0]←Z←Z-4;                   00003250    ARC01450
                     IG←0; IF K=B THEN                                      00003260    ARC01460
                      BEGIN                                                 00003270    ARC01460
                        L←1; FORJ Z DO WGRP[1,J]←WGRP[0,J];                 00003280    ARC01460
                        GO TO LENDK                                         00003290    ARC01460
                      END;                                                  00003300    ARC01460
                     FORI1 L DO FORJ1 WGRP[I,0] DO FOR P←1 STEP 1 UNTIL     00003310    ARC01470
                     WGRP[0,0] DO IF WGRP[I,J]=WGRP[0,P] THEN               00003320    ARC01480
                      BEGIN                                                 00003330    ARC01480
                        IF IG=0 THEN                                        00003340    ARC01490
                         BEGIN                                              00003350    ARC01490
                           IG←I; X←0                                        00003360    ARC01490
                         END ELSE X←I; Z←WGRP[IG,0];                        00003370    ARC01490
                        Y←WGRP[X,0];                                        00003380    ARC01490
                        FOR KA←1 STEP 1 UNTIL P-1 DO WGRP[IG,KA+Z]←WGRP[    00003390    ARC01500
                        X,KA]; KB←Z+P-1;                                    00003400    ARC01500
                        FOR P←P+1 STEP 1 UNTIL Y DO                         00003410    ARC01510
                         BEGIN                                              00003420    ARC01510
                           FOR KA←1 STEP 1 UNTIL Z DO IF WGRP[IG,KA]=       00003430    ARC01520
                           WGRP[X,P] THEN KA←99;                            00003440    ARC01520
                           IF KA<99 THEN WGRP[IG,KB←KB+1]←WGRP[X,P]         00003450    ARC01530
                         END;                                               00003460    ARC01530
                        WGRP[IG,0]←KB;                                      00003470    ARC01540
                        IF X≠0 THEN WGRP[X,0]←0; R←J←NDM;                   00003480    ARC01560
                      END IF WGRP AND FORI;                                 00003490    ARC01560
                     IF IG=0 THEN                                           00003500    ARC01570
                      BEGIN                                                 00003510    ARC01570
                        L←L+1;                                              00003520    ARC01570
                        FORJ WGRP[0,0] DO WGRP[L,J]←WGRP[0,J];              00003530    ARC01570
                      END;                                                  00003540    ARC01570
LENDK:              END FOR K LOOP;                                         00003550    ARC01580
                    K←0; FORI NDM DO GROUPS[I]←0;                           00003560    ARC01590
                    FORI1 L DO                                              00003570    ARC01600
                     BEGIN                                                  00003580    ARC01600
                       P←WGRP[I,0]; IF P≠0 THEN                             00003590    ARC01610
                        BEGIN                                               00003600    ARC01610
                          K←K+1;                                            00003610    ARC01610
                          FORJ1 P DO GROUPS[WGRP[I,J]]←K;                   00003620    ARC01610
                        END;                                                00003630    ARC01610
                     END;                                                   00003640    ARC01610
                    FOR I←B STEP 1 UNTIL E DO ALTERS[I,11]←GROUPS[ALTERS[    00003650    ARC01620
                    I,5].[30:12]]; NUMGRPS←K;                               00003660    ARC01630
```

```
                END FINDGROUPS;                                                 00003670        ARC01630
                                                                                00003680        ARC01631
COMMENT ATKTYPE UPDATES TYPATK (TYPE OF ATTACK LIST) GIVEN THAT TARGET T00003690        ARC01631
        IS UNDER ATTACK IN PERIOD P;                                            00003700        ARC01632
PROCEDURE       ATKTYPE(T,P);                                                   00003710        ARC01640
VALUE           T,P;                                                            00003720        ARC01640
INTEGER         T,P;                                                            00003730        ARC01640
                BEGIN                                                           00003740        ARC01640
INTEGER           I,J,KA;                                                       00003750        ARC01640
                  UNPACK(TGTTAD[T,1],ACTGT[0,*]);                               00003760        ARC01650
                  FORI 7 DO                                                     00003770        ARC01660
                   BEGIN                                                        00003780        ARC01660
                     KA←ACTGT[0,I];                                             00003790        ARC01660
                     IF KA≠" " THEN FORJ 7 DO IF KA=ACTGT[1,J] THEN             00003800        ARC01670
                      BEGIN                                                     00003810        ARC01670
                        ACTGT[2,J]←P; J←7                                       00003820        ARC01670
                      END                                                       00003830        ARC01670
                   END;                                                         00003840        ARC01670
                  FORJ 7 DO ACTGT[0,J]←0; J←0;                                  00003850        ARC01680
                  FORI 7 DO IF ACTGT[2,I]>P-ACTIVEINDEX THEN                    00003860        ARC01680
                   BEGIN                                                        00003870        ARC01680
                     ACTGT[0,J]←ACTGT[ 1,1]; J←J+1                              00003880        ARC01690
                   END;                                                         00003890        ARC01690
                  PACKWORD(TYPATK,ACTGT[0,*]);                                  00003900        ARC01690
                END ATKTYPE;                                                    00003910        ARC01690
                                                                                00003920        ARC01691
COMMENT COMBOUT WRITES OUT DM AND TDM INFO STORED IN TABLE ARR;                 00003930        ARC01691
PROCEDURE       COMBOUT(ARR,N);                                                 00003940        ARC01700
VALUE           N;                                                              00003950        ARC01700
INTEGER         N;                                                              00003960        ARC01700
INTEGER ARRAY   ARR[0];                                                         00003970        ARC01700
                BEGIN                                                           00003980        ARC01700
INTEGER           I,K,KA,KB,KC,KD;                                              00003990        ARC01710
                  KA←(IF N=1 THEN FINALT ELSE LASTALT); K←-1;                   00004000        ARC01720
                  WRITE(DATAOUT [DBL]);.                                        00004010        ARC01730
                  FORI KA DO                                                    00004020        ARC01740
                   BEGIN                                                        00004030        ARC01740
                     BREAKDMWORD(ARR[I],KB,KB,KC,KD);                           00004040        ARC01740
                     IF KB=0 THEN                                               00004050        ARC01750
                      BEGIN                                                     00004060        ARC01750
                        INTC[K←K+1]←0;                                          00004070        ARC01750
                        INTC[K←K+1]←ARR[I]                                      00004080        ARC01750
                      END ELSE                                                  00004090        ARC01750
                      BEGIN                                                     00004100        ARC01750
                        INTC[K←K+1]←KC; INTC[K←K+1]←KD                          00004110        ARC01760
                      END;                                                      00004120        ARC01760
                   END;                                                         00004130        ARC01760
                  IF N=1 THEN WRITE(DATAOUT,LBEST,FORI K DO INTC[I])            00004140        ARC01770
                  ELSE                                                          00004150        ARC01770
                   BEGIN                                                        00004160        ARC01770
                     WRITE(DATAOUT,BESTC,FORI K DO INTC[I]);                    00004170        ARC01780
                     K←2×(FUTALT-1)+1; KB←-1;                                   00004180        ARC01790
                     FORI FUTALT-1 DO                                           00004190        ARC01790
                      BEGIN                                                     00004200        ARC01790
                        KA←KB+1; KB←KA+1;                                       00004210        ARC01790
                        IF BESTALT[I]=0 THEN INTC[KA]←INTC[KB]←0;               00004220        ARC01800
                      END;                                                      00004230        ARC01800
                     WRITE(DATAOUT,BESTA,FORI K DO INTC[I]);                    00004240        ARC01810
                   END                                                          00004250        ARC01810
                END COMBOUT;                                                    00004260        ARC01820
                                                                                00004270        ARC01821
```

```
COMMENT MULT CALCULATES THE SUM OF THE PRODUCTS OF N ELEMENTS IN ARRAYS 00004280  ARC01821
        A AND B AND STORES THE RESULT IN WORD C, I IS THE ITERATION VAR;00004290  ARC01822
PROCEDURE       MULT(A,B,C,I,N);                                        00004300  ARC01830
VALUE           N;                                                      00004310  ARC01830
ARRAY           A,B[0];                                                 00004320  ARC01830
REAL            C;                                                      00004330  ARC01830
INTEGER         N,I;                                                    00004340  ARC01830
                 BEGIN                                                  00004350  ARC01830
                   C←0; FORI N DO C←C+A[I]×B[I];                        00004360  ARC01840
                 END MULT;                                              00004370  ARC01840
PROCEDURE       GENTARGETS;                                             00004380  ARC01850
                 BEGIN                                                  00004390  ARC01850
        COMMENT READ DATA,GENERATE AND SORT TGTLCV,TGTTAD AND TGTMNX;   00004400  ARC01860
LABEL             L1,L2,L3;                                             00004410  ARC01870
DEFINE            LOADSQRTGT=KE←SQRTGT[K,J,0]←SQRTGT[K,J,0]+1;          00004420  ARC01880
                  SQRTGT[K,J,KE]←I #; I←0; T←0; TA←0;                   00004430  ARC01890
L1;               I←I+1; READ(DATAIN [NO],A,CODE)[L2];                  00004440  ARC01910
                  IF CODE≠ "T   " THEN GO TO L2;                        00004450  ARC01910
                  READ(DATAIN ,TGTIN,CODE,FORJ 5 DO TGTTAD[I,J],FORK 4  00004460  ARC01920
                  DO TGTLCV[I,K]); INTA[I]←TGTLCV[I,0];                 00004470  ARC01930
                  TB←TGTLCV[I,2]-TGTLCV[I,0];                           00004480  ARC01930
                  IF T<TB THEN T←TB;                                    00004490  ARC01930
                  TOTVALUE←TOTVALUE+TGTLCV[I,4];                        00004500  ARC01940
                  UNPACK(TGTTAD[I,1],INTC); KA←ACTGT[0,0];              00004510  ARC01950
                  FORJ 7 DO                                             00004520  ARC01950
                   BEGIN                                                00004530  ARC01950
                     KB←INTC[J];                                        00004540  ARC01950
                     IF KB≠" " AND KB≠0 THEN IF KA+1=OCCUR(ACTGT[1,*],  00004550  ARC01970
                     KA+1,KB,8,8,1) THEN                                00004560  ARC01970
                      BEGIN                                             00004570  ARC01970
                        ACTGT[1,KA]←KB; KA←KA+1                         00004580  ARC01980
                      END                                               00004590  ARC01980
                   END;                                                 00004600  ARC01980
                  ACTGT[0,0]←KA; NTYPT←KA; GO TO L1;                    00004610  ARC01990
L2;               KA←I-1; FORJ1 KA DO                                   00004620  ARC02000
                   BEGIN                                                00004630  ARC02000
                     TA←-1;                                             00004640  ARC02000
                     FORI1 KA DO IF TA< INTA[I] THEN                    00004650  ARC02010
                      BEGIN                                             00004660  ARC02010
                        TA←INTA[I]; K←I;                                00004670  ARC02010
                      END;                                              00004680  ARC02010
                     TGTMNX[J]←INTA[K]; INTC[J]←K ;                     00004690  ARC02020
                     INTA[K]←-1;                                        00004700  ARC02020
                   END;                                                 00004710  ARC02020
                  NTGT←KA; FORT1 NTGT DO                                00004720  ARC02040
                   BEGIN                                                00004730  ARC02040
                     KA←INTC[I]; IF I=KA THEN GO TO L3;                 00004740  ARC02050
                     FORJ 4 DO                                          00004750  ARC02060
                      BEGIN                                             00004760  ARC02060
                        TA←TGTLCV[I,J];                                 00004770  ARC02060
                        TGTLCV[I,J]←TGTLCV[KA,J];                       00004780  ARC02060
                        TGTLCV[KA,J]←TA;                                00004790  ARC02070
                      END;                                              00004800  ARC02070
                     FORJ 5 DO                                          00004810  ARC02080
                      BEGIN                                             00004820  ARC02080
                        K←TGTTAD[I,J];                                  00004830  ARC02080
                        TGTTAD[I,J]←TGTTAD[KA,J];                       00004840  ARC02080
                        TGTTAD[KA,J]←K;                                 00004850  ARC02090
                      END;                                              00004860  ARC02090
                     FOR J←I STEP 1 UNTIL NTGT DO IF I=INTC[J] THEN     00004870  ARC02100
                      BEGIN                                             00004880  ARC02100
```

```
                    INTC[J]←KA; GO TO L3                                    00004890    ARC02110
                  END;                                                      00004900    ARC02110
L3:             END;                                                        00004910    ARC02120
                FOR I1 NTGT DO                                              00004920    ARC02130
                 BEGIN                                                      00004930    ARC02130
                   KA←ENTIER(TGTLCV[I,0]);                                  00004940    ARC02130
                   KB←ENTIER(TGTLCV[I,2]);                                  00004950    ARC02130
                   KC←ENTIER(TGTLCV[I,1]);                                  00004960    ARC02140
                   KD←ENTIER(TGTLCV[I,3]); K←KA; J←KC;                      00004970    ARC02150
                   LOADSQRTGT; IF KA≠KB THEN                                00004980    ARC02150
                    BEGIN                                                   00004990    ARC02150
                      K←KB; LOADSQRTGT;                                     00005000    ARC02150
                    END;                                                    00005010    ARC02150
                   IF KC≠KD THEN                                            00005020    ARC02160
                    BEGIN                                                   00005030    ARC02160
                      K←KA; J←KD; LOADSQRTGT;                               00005040    ARC02160
                      IF KA≠KB THEN                                         00005050    ARC02160
                       BEGIN                                                00005060    ARC02160
                         K←KB; LOADSQRTGT;                                  00005070    ARC02170
                       END;                                                 00005080    ARC02170
                    END;                                                    00005090    ARC02170
                 END FOR I1 LOOP;                                           00005100    ARC02170
                TGTMNX[0] ←T; GO TO REPORTS;                                00005110    ARC02180
              END PROCEDURE GENTARGETS;                                     00005120    ARC02190
PROCEDURE     GENSQRDTA;                                                    00005130    ARC02200
              BEGIN                                                         00005140    ARC02200
      COMMENT GENERATE SQRDTA;                                              00005150    ARC02210
INTEGER       I,J,K,KA,KB;                                                  00005160    ARC02220
REAL          DMX,DMY,MR,RX,RY;                                             00005170    ARC02220
LABEL         L1;                                                           00005180    ARC02220
              FOR I NDM DO                                                  00005190    ARC02230
               BEGIN                                                        00005200    ARC02230
                 MR←0;                                                      00005210    ARC02230
                 FOR K TYPE-1 DO IF MR<DMSPEC[2×K] THEN IF DMLOCO[I,        00005220    ARC02240
                 K+3]≠0 THEN MR←DMSPEC[2×K]; INTA[I]←MR;                    00005230    ARC02240
               END;                                                         00005240    ARC02240
              FOR J NSQR DO                                                 00005250    ARC02250
               BEGIN                                                        00005260    ARC02250
                 KB←MSQR+2;                                                 00005270    ARC02250
                 FOR I MSQR DO                                              00005280    ARC02250
                  BEGIN                                                     00005290    ARC02250
                    SQRDTA[J,I]←KA←KB;                                      00005300    ARC02250
                    FOR K NDM DO                                            00005310    ARC02260
                     BEGIN                                                  00005320    ARC02260
                       DMX←DMLOCO[K,0]; DMY←DMLOCO[K,1];                    00005330    ARC02270
                       MR←INTA[K]; RX←DMX-J; RY←DMY-I;                      00005340    ARC02270
                       IF MR≤-RX OR MR≤RX-1 THEN GO TO L1;                  00005350    ARC02280
                       IF MR≤-RY OR MR≤RY-1 THEN GO TO L1;                  00005360    ARC02290
                       IF (.7071+MR)*2 ≤(RX-.5)*2 +(RY-.5)*2 THEN           00005370    ARC02300
                       GO TO L1; SQRDTA[J,KB]←K; KB←KB+1;                   00005380    ARC02310
L1:                   END;                                                  00005390    ARC02320
                    IF KB=KA THEN KB←KB+1;                                  00005400    ARC02330
                  END;                                                      00005410    ARC02330
                 SQRDTA[J,MSQR+1]←KB;                                       00005420    ARC02340
               END;                                                         00005430    ARC02340
              END PROCEDURE GENSQRDATA;                                     00005440    ARC02350
PROCEDURE     GENATTACK;                                                    00005450    ARC02360
              BEGIN                                                         00005460    ARC02360
      COMMENT GENERATE  ATTACK;                                             00005470    ARC02370
                FOR J1 @10 DO                                               00005480    ARC02370
                 BEGIN                                                      00005490    ARC02370
```

```
                    READ(DATAIN [NO],A,CODE)[PROCESS];                    00005500  ARC02380
                    IF CODE≠"AM " THEN GO TO REPORTS;                     00005510  ARC02380
                    READ(DATAIN,/,I,FORI 6 DO ATTACK[J,I]);               00005520  ARC02390
                    NAM←J; ATTACK[J,7]←1;                                 00005530  ARC02390
                  END;                                                    00005540  ARC02390
                END PROCEDURE GENATTACK;                                  00005550  ARC02400
PROCEDURE       GENEWATT;                                                 00005560  ARC02410
                BEGIN                                                     00005570  ARC02410
      COMMENT GENERATE NEWATTACK FOR BOOK GAME RUN;                       00005580  ARC02420
LABEL             FIN;                                                    00005590  ARC02420
                  NAM←0;                                                  00005600  ARC02430
                  FORJ1 999 DO                                            00005610  ARC02430
                   BEGIN                                                  00005620  ARC02430
                     READ(DATAIN[NO],A,CODE)[FIN];                        00005630  ARC02440
                     IF CODE≠"AM " THEN GO TO FIN;                        00005640  ARC02440
                     READ(DATAIN,/,I,FORI 6 DO ATTACK[J,I])               00005650  ARC02450
                   END FORJ;                                              00005660  ARC02450
FIN;              NAM←J-1                                                 00005670  ARC02460
                END GENEWATT;                                             00005680  ARC02460
                                                                          00005690  ARC02461
COMMENT GENTDMTGT BUILDS TABLE TDMTGT;                                    00005700  ARC02461
PROCEDURE       GENTDMTGT;                                                00005710  ARC02470
                BEGIN                                                     00005720  ARC02470
                  FORI NTGT DO INTC[I]←-1;                                00005730  ARC02480
                  FORI1 NTGT DO FOR J←2,3,4,5 DO                          00005740  ARC02490
                   BEGIN                                                  00005750  ARC02490
                     KA←TGTTAD [I,J];                                     00005760  ARC02490
                     IF KA≠0 THEN                                         00005770  ARC02500
                      BEGIN                                               00005780  ARC02500
                        KB←INTC[KA]←1+INTC[KA];                           00005790  ARC02500
                        TDMTGT[KA,KB]←I;                                  00005800  ARC02500
                      END                                                 00005810  ARC02500
                   END                                                    00005820  ARC02500
                END GENTDMTGT;                                            00005830  ARC02510
                                                                          00005840  ARC02511
COMMENT OPENROW CREATES NECESSARY  FLAGS TO START AN OVERFLOW ROW FOR AN   00005850  ARC02511
        AM IN TABLE  AMDATA, IT IS USED ONLY BY GENAMDATA;                00005860  ARC02512
PROCEDURE       OPENROW(R,KA,FULL);                                       00005870  ARC02520
LABEL           FULL;                                                     00005880  ARC02530
INTEGER         KA,R;                                                     00005890  ARC02530
                BEGIN                                                     00005900  ARC02530
LABEL             L1;                                                     00005910  ARC02530
INTEGER           I,J,RA;                                                 00005920  ARC02530
                  FOR J←R+1 STEP 1 UNTIL MAMDATA DO IF AMDATA[J,0]=0      00005930  ARC02540
                  THEN                                                    00005940  ARC02540
                   BEGIN                                                  00005950  ARC02540
                     RA←J; GO TO L1                                       00005960  ARC02550
                   END;                                                   00005970  ARC02550
                  GO TO FULL;                                             00005980  ARC02550
L1;               AMDATA[R,7]←RA; AMDATA[RA,0]←-R;                        00005990  ARC02560
                  AMDATA[R,8]←KA;                                         00006000  ARC02560
                  FOR J←1 STEP 1 UNTIL 5 DO AMDATA[RA,J]←AMDATA[R,J];     00006010  ARC02570
                  KA←8; R←RA                                              00006020  ARC02570
                END;                                                      00006030  ARC02570
                                                                          00006040  ARC02571
COMMENT FINDTRGT IS THE INDEX TO TGTTAD FOR THE TARGET LOCATED AT X,Y;    00006050  ARC02571
INTEGER PROCEDURE FINDTRGT(X,Y);                                          00006060  ARC02580
VALUE           X,Y;                                                      00006070  ARC02580
REAL            X,Y;                                                      00006080  ARC02580
                BEGIN                                                     00006090  ARC02580
INTEGER           I,K;                                                    00006100  ARC02590
```

```
REAL            XB,DEL;                                                  00006110  ARC02590
LABEL           FIN1,FIN,L1, L3,L4;                                      00006120  ARC02590
                IF X< TGTMNX[NTGT] THEN GO TO FIN1;                      00006130  ARC02600
                DEL←NTGT; K←1; XB←X-TGTMNX[0];                           00006140  ARC02610
                IF X≥TGTMNX[1] THEN GO TO L4;                            00006150  ARC02620
L1:             DEL←.5×DEL; K←K+DEL; GO TO L3 ;                          00006160  ARC02630
L3:             IF DEL<1 THEN GO TO L4;                                  00006170  ARC02640
                IF X<TGTMNX[K] THEN GO TO L1; DEL←.5×DEL;                00006180  ARC02640
                K←K-DEL; GO TO L3;                                       00006190  ARC02650
L4:             FOR I←K STEP 1 UNTIL NTGT DO                             00006200  ARC02660
                 BEGIN                                                   00006210  ARC02660
                   IF XB>TGTLCV[I,0] THEN GO TO FIN1;                    00006220  ARC02670
                   IF X≥TGTLCV[I,0] AND X≤TGTLCV[I,2] THEN IF Y≥         00006230  ARC02680
                   TGTLCV[I,1] AND Y≤TGTLCV[I,3] THEN                    00006240  ARC02680
                    BEGIN                                                00006250  ARC02680
                      FINDTRGT←I; GO TO FIN                              00006260  ARC02690
                    END                                                  00006270  ARC02690
                 END;                                                    00006280  ARC02690
FIN1:           NOTGT←TRUE;                                              00006290  ARC02690
FIN:           END FINT;                                                 00006300  ARC02690
                                                                         00006310  ARC02691
COMMENT GENAMDATA DETERMINES FOR EACH PERIOD THE BM WHICH MAY ENGAGE AN  00006320  ARC02691
        AM AND BUILDS TABLE AMDATA;                                      00006330  ARC02692
PROCEDURE       GENAMDATA(TM,X1,Y1,X2,Y2,SP,IDENT,FULL,P);               00006340  ARC02710
VALUE           TM,X1,Y1,X2,Y2,SP,IDENT,P;                               00006350  ARC02720
INTEGER         IDENT;                                                   00006360  ARC02720
BOOLEAN         P;                                                       00006370  ARC02720
REAL            TM,X1,X2,Y1,Y2,SP;                                       00006380  ARC02730
LABEL           FULL;                                                    00006390  ARC02730
                 BEGIN                                                   00006400  ARC02730
REAL              S,INT,MINX,MAXX,MINXY,MAXXY,YA,A,B,C,DEL,TGZ;         00006410  ARC02740
INTEGER           I,R,K,XS,YS,L,J,KA,KB,KC,KD,LA,LB,LP,RA;              00006420  ARC02750
LABEL             FIN,L1,L2,L3;                                          00006430  ARC02760
BOOLEAN           NOS;                                                   00006440  ARC02760
                  NODEFENSE←NOTGT←TOOLATE←FALSE;                         00006450  ARC02770
                  IF NOT SOOK THEN GENSORDTA;                            00006460  ARC02780
                  IF NOT P THEN                                          00006470  ARC02790
                   BEGIN                                                 00006480  ARC02790
                     K←FINDTRGT(X2,Y2);                                  00006490  ARC02800
                     IF NOTGT THEN GO TO FIN;                            00006500  ARC02800
                   END;                                                  00006510  ARC02800
                  FOR I←1 STEP 1 UNTIL MAMDATA DO IF AMDATA[I,0]=0 THEN  00006520  ARC02810
                   BEGIN                                                 00006530  ARC02810
                     R←LASTPROW← I; GO TO L1                             00006540  ARC02820
                   END;                                                  00006550  ARC02820
                  GO TO FULL;                                            00006560  ARC02820
L1:               AMDATA[R,0]←1; AMDATA[R,1]←IDENT;                      00006570  ARC02830
                  TGZ←SQRT((Y1-Y2)×(Y1-Y2)+(X1-X2)×(X1-X2))/SP+TM;       00006580  ARC02840
                  LP←AMDATA[R,2]←ENTIER(TGZ); AMDATA[R,5]←9;             00006590  ARC02850
                  AMDATA[R,7]←0;                                         00006600  ARC02850
                  IF LP>MAXPERIOD THEN MAXPERIOD←LP;                     00006610  ARC02860
                  IF NOT P THEN                                          00006620  ARC02870
                   BEGIN                                                 00006630  ARC02870
                     AMDATA[R,3]←K;                                      00006640  ARC02870
                     FOR I←2 STEP 1 UNTIL 5 DO IF TOLTSTTGTTAD[K,I]≠0    00006650  ARC02880
                     THEN                                                00006660  ARC02880
                      BEGIN                                              00006670  ARC02880
                        AMDATA[R,4]←TGTTAD[K,0];                         00006680  ARC02890
                        GO TO L2                                         00006690  ARC02890
                      END;                                               00006700  ARC02890
                     AMDATA[R,4]←-TGTTAD[K,0];                           00006710  ARC02890
```

```
 END;                                                                  00006720  ARC02890
L2:  IF NOT P THEN WRITE(DATAOUT,<"AM",I4,                             00006730  ARC02900
     " ENTERED SYSTEM AT (",F8.3,",",F8.3,").  TARGET IS "             00006740  ARC02920
     ,A6,"  DURING PERIOD",I4>,IDENT,X1,Y1,TGTTAB[K,0],LP);            00006750  ARC02920
     IF NOT P AND HEURISTICS THEN IF TGTLOV[K,4]≤MINTGTVAL             00006760  ARC02930
     THEN                                                              00006770  ARC02930
      BEGIN                                                            00006780  ARC02930
        NOTGT←TRUE;                                                    00006790  ARC02940
        WRITE(DATAOUT,<                                                00006800  ARC02940
        "TARGET VALUE BELOW MINIMUM, AM IGNORED">);                    00006810  ARC02940
        AMDATA[R,0]←0; GO TO FIN                                       00006820  ARC02950
      END;                                                             00006830  ARC02950
     IF LP<PERIOD THEN                                                 00006840  ARC02960
      BEGIN                                                            00006850  ARC02960
        TOOLATE←TRUE; LASTPROW←R; AMDATA[R,0]←0;                       00006860  ARC02970
        GO TO FIN;                                                     00006870  ARC02970
      END;                                                             00006880  ARC02970
     IF (X2-X1)=0 THEN NOS←TRUE ELSE                                   00006890  ARC02980
      BEGIN                                                            00006900  ARC02980
        NOS←FALSE; S←(Y2-Y1)/(X2-X1);                                  00006910  ARC02990
        INT←Y1-S×X1;                                                   00006920  ARC02990
        IF ABS(S)>10 THEN NOS←TRUE                                     00006930  ARC02990
      END;                                                             00006940  ARC02990
     IF X1<X2 THEN                                                     00006950  ARC03000
      BEGIN                                                            00006960  ARC03000
        MINX←X1; MAXX←X2; MINXY←Y1;                                    00006970  ARC03000
        MAXXY←Y2                                                       00006980  ARC03000
      END ELSE                                                         00006990  ARC03000
      BEGIN                                                            00007000  ARC03000
        MINX←X2; MAXX←X1; MINXY←Y2;                                    00007010  ARC03010
        MAXXY←Y1                                                       00007020  ARC03010
      END;                                                             00007030  ARC03010
     XS←ENTIER(MINX); YS←ENTIER(MINXY);                                00007040  ARC03020
     SQRX[0]←XS; SQRY[0]←YS; L←0;                                      00007050  ARC03020
     IF MINXY≤MAXXY THEN I←1 ELSE I←-1;                                00007060  ARC03030
L3:  IF(XS+1)≤MAXX THEN YA←S×(XS+1)+INT ELSE YA←ENTIER(                 00007070  ARC03040
     MAXXY);                                                           00007080  ARC03040
     WHILE I×(YS←YS+I)≤YA×I DO                                         00007090  ARC03060
      BEGIN                                                            00007100  ARC03060
        L←L+1; SQRX[L]←XS; SQRY[L]←YS                                  00007110  ARC03060
      END;                                                             00007120  ARC03060
     XS←XS+1; IF XS≤MAXX THEN                                          00007130  ARC03070
      BEGIN                                                            00007140  ARC03070
        YS←YS-2×I; GO TO L3;                                           00007150  ARC03070
      END;                                                             00007160  ARC03070
     LB←-1; FORI MTGT DO INTC[I]←0;                                    00007170  ARC03080
     FOR I←0 STEP 1 UNTIL L DO                                         00007180  ARC03090
      BEGIN                                                            00007190  ARC03090
                    COMMENT DETERMINE DM-AM TRAJ INTERSCT;             00007200  ARC03090
        NEXT,NONE,L4;                                                  00007210  ARC03100
LABEL   KA←SQRX[I]; J←SQRY[I]; KB←SQRDTA[KA,J];                        00007220  ARC03110
        KC←SQRDTA[KA,J+1]-1;                                           00007230  ARC03110
        FOR J←KB STEP 1 UNTIL KC DO                                    00007240  ARC03120
         BEGIN                                                         00007250  ARC03120
           KD←SQRDTA[KA,J];                                            00007260  ARC03120
           IF KD=0 THEN GO TO NEXT;                                    00007270  ARC03130
           FORK NTYPE=1 DO IF NOT CHEKBIT(DMVOID[K],KD)                00007280  ARC03140
           THEN GO TO L4; GO TO NEXT;                                  00007290  ARC03140
L4:        IF OCCUR(INTC,LB+1,KD,6,7,2) ≠LB+1 THEN GO TO               00007300  ARC03150
           NEXT;                                                       00007310  ARC03150
           IF NOS THEN                                                 00007320  ARC03160
```

```
                    BEGIN                                                00007330        ARC03160
                      S←(X2-X1)/(Y2-Y1); INT←X1-S×Y1;                    00007340        ARC03160
                      MINX←DMLOCO[KD,1];                                 00007350        ARC03160
                      YA←DMLOCO[KD,0]                                    00007360        ARC03170
                    END ELSE                                             00007370        ARC03170
                    BEGIN                                                00007380        ARC03170
                      MINX←DMLOCO[KD,0];                                 00007390        ARC03180
                      YA←DMLOCO[KD,1];                                   00007400        ARC03180
                    END;                                                 00007410        ARC03180
                   A←(1+S×S); B←2×(S×(INT-YA)-MINX);                     00007420        ARC03190
                   C←INT×INT-2×INT×YA+MINX×MINX+YA×YA;                   00007430        ARC03200
                   MINX←2×A; YA←0;                                       00007440        ARC03200
                   FOR LA←0 STEP 1 UNTIL NTYPE-1 DO                      00007450        ARC03210
                    BEGIN                                                00007460        ARC03210
                      IF CHEKBIT(DMVOID[LA],KD) THEN GO TO NONE;         00007470        ARC03220
                      C←C+YA; YA←DMSPEC[2×LA ]*2;                        00007480        ARC03230
                      C←C-YA; MAXX← B×B-4×A×C;                           00007490        ARC03240
                      IF MAXX<0 THEN GO TO NONE;                         00007500        ARC03240
                      MAXX←SQRT(MAXX); LB←LB+1;                          00007510        ARC03250
                      INTA[LB]←(-B+MAXX)/MINX;                           00007520        ARC03260
                      INTB[LB]←(-B-MAXX)/MINX;                           00007530        ARC03260
                      INTC[LB]←LA & KD[30:36:12];                        00007540        ARC03270
NONE:               END;                                                 00007550        ARC03280
NEXT:             END;                                                   00007560        ARC03290
                 END;                                                    00007570        ARC03290
                IF LB<0 THEN IF AMDATA[R,4]>0 THEN                       00007580        ARC03300
                 BEGIN                                                   00007590        ARC03300
                   AMDATA[R,A]←AMDATA[R,7]←0;                            00007600        ARC03300
                   AMDATA[R,8]←8;                                        00007610        ARC03310
                   GO TO FIN                                             00007620        ARC03310
                 END ELSE                                                00007630        ARC03310
                 BEGIN                                                   00007640        ARC03310
                   NODEFENSE←TRUE; LASTPROW←R;                           00007650        ARC03310
                   WRITE(DATAOUT,<"AM",I4," NOT COVERED,TARGET IS ",     00007660        ARC03320
                   A6,"DURING PERIOD",I4>,IDENT,AMDATA[R,4],AMDATA[R,    00007670        ARC03330
                   2]); AMDATA[R,0]←0; GO TO FIN;                        00007680        ARC03330
                 END;                                                    00007690        ARC03330
                DEL←SQRT(SP×SP/(S×S+1)) ;                                00007700        ARC03340
                IF NOS THEN                                              00007710        ARC03350
                 BEGIN                                                   00007720        ARC03350
                   IF Y2<Y1 THEN DEL←-DEL                                00007730        ARC03350
                 END ELSE IF X2<X1 THEN DEL←-DEL;                        00007740        ARC03360
                YA← DEL×(PERIOD-TM-1)+(IF NOS THEN Y1 ELSE X1);          00007750        ARC03370
                KA←LA+9;                                                 00007760        ARC03370
                FOR I←PERIOD STEP 1 UNTIL LP DO                          00007770        ARC03380
                 BEGIN                                                   00007780        ARC03380
                                    COMMENT SET UP AMDATA ENTRIES;       00007790        ARC03380
                   IF LA≠KA THEN                                         00007800        ARC03390
                    BEGIN                                                00007810        ARC03390
                      IF KA=NAMDATA THEN OPENROW(R,KA,FULL);             00007820        ARC03390
                      IF KA=NAMDATA-1 THEN                               00007830        ARC03400
                       BEGIN                                             00007840        ARC03400
                         AMDATA[R,KA+1]←0;                               00007850        ARC03400
                         OPENROW(R,KA,FULL);                             00007860        ARC03400
                       END;                                              00007870        ARC03400
                      LA←KA+1;                                           00007880        ARC03410
                    END;                                                 00007890        ARC03410
                   KA←LA; A←B←YA←YA+DEL;                                 00007900        ARC03410
                   IF I=LP THEN DEL←SIGN(DEL)×SQRT((SP×(TGT-LP))*2/(S×   00007910        ARC03420
                   S+1));                                                00007920        ARC03420
                   IF DEL<0 THEN A←A+DEL ELSE B←B+DEL;                   00007930        ARC03430
```

```
                    FOR L←0 STEP 1 UNTIL LB DO                                  00007940  ARC03440
                     BEGIN                                                      00007950  ARC03440
                       IF A≤INTA[L] AND B>INTB[L] THEN                          00007960  ARC03450
                        BEGIN                                                   00007970  ARC03450
                          AMDATA[R,6]←I; AMDATA[R,LA]←-I;                       00007980  ARC03460
                          KA←KA+1;                                              00007990  ARC03460
                          AMDATA[R,KA]←INTC[L] & I[18:36:12];                   00008000  ARC03460
                        END;                                                    00008010  ARC03460
                       IF KA=NAMDATA THEN OPENROW(R,KA,FULL)                    00008020  ARC03470
                     END                                                        00008030  ARC03470
                   END;                                                         00008040  ARC03470
                  IF LA=KA THEN KA←KA-1; AMDATA[R,KA+1]←-LP;                    00008050  ARC03480
                  AMDATA[R,8 ]←KA;                                              00008060  ARC03480
FIN:              IF MAXAMDT<R THEN MAXAMDT←R;                                  00008070  ARC03500
                END PROCEDURE GENAMDATA;                                        00008080  ARC03500
                                                                                00008090  ARC03501
COMMENT GENDMSPEC READS DM RANGE AND EFFECTIVENESS AND BUILDS DMSPEC;           00008100  ARC03501
PROCEDURE       GENDMSPEC;                                                      00008110  ARC03520
                BEGIN                                                           00008120  ARC03520
                  FORK @10 DO                                                   00008130  ARC03520
                   BEGIN                                                        00008140  ARC03520
                     READ(DATAIN [NO],A,CODE)[PROCESS];                         00008150  ARC03530
                     IF CODE≠"SP " THEN GO TO REPORTS;                          00008160  ARC03540
                     READ(DATAIN,/,I,I,TA,TB);                                  00008170  ARC03540
                     DMSPEC[2×I]←TA; DMSPEC[2×I+1]←TB;                          00008180  ARC03550
                   END;                                                         00008190  ARC03550
                END DMSPEC;                                                     00008200  ARC03550
                                                                                00008210  ARC03551
COMMENT GENTDLIST READS TDM INFO AND BUILDS TDMLIST;                            00008220  ARC03551
PROCEDURE       GENTDLIST;                                                      00008230  ARC03560
                BEGIN                                                           00008240  ARC03560
                  FORJ @10 DO                                                   00008250  ARC03570
                   BEGIN                                                        00008260  ARC03570
                     READ(DATAIN [NO],A,CODE)[PROCESS];                         00008270  ARC03570
                     IF CODE≠"TDM" THEN GO TO REPORTS;                          00008280  ARC03580
                     READ(DATAIN,/,K,K,TDLIST[K])                               00008290  ARC03580
                   END;                                                         00008300  ARC03580
                END PROCEDURE GENTDLIST;                                        00008310  ARC03590
                                                                                00008320  ARC03591
COMMENT GENDMLOCO READS DM SITE INFO AND BUILDS DMLOCO;                         00008330  ARC03591
PROCEDURE       GENDMLOCO;                                                      00008340  ARC03600
                BEGIN                                                           00008350  ARC03600
                  FORK @10 DO                                                   00008360  ARC03600
                   BEGIN                                                        00008370  ARC03600
                     READ(DATAIN [NO],A,CODE)[PROCESS];                         00008380  ARC03610
                     IF CODE≠"DM " THEN GO TO REPORTS;                          00008390  ARC03610
                     READ(DATAIN,/,I,I,FOR J←0,1,3,4 DO DMLOCO[I,J]);           00008400  ARC03620
                     DMLOCO[I,2]←DMLOCO[I,3]←DMLOCO[I,4];                       00008410  ARC03630
                     IF NDM<I THEN NDM←I;                                       00008420  ARC03630
                     FORK NTYPE-1 DO IF DMLOCO[I,K+3]≠0 THEN CLEARBIT(          00008430  ARC03640
                     DMVOID[K],I);                                              00008440  ARC03640
                   END;                                                         00008450  ARC03640
                END PROCEDURE GENDMLOCO;                                        00008460  ARC03650
PROCEDURE       PLAUSIBLE;                                                      00008470  ARC03660
                BEGIN                                                           00008480  ARC03660
                                COMMENT GENERATE FACDTA AND PACUMS;             00008490  ARC03660
INTEGER           CT,CU,X1,X2,Y1,Y2;                                            00008500  ARC03670
BOOLEAN           C;                                                            00008510  ARC03670
LABEL             L3,L4,L5,FIN,FINISH,L6;                                       00008520  ARC03680
INTEGER ARRAY     T,UT0:7];                                                     00008530  ARC03680
                  PACIN←TRUE;                                                   00008540  ARC03690
```

```
FORI 810 DO                                                                    00008550    ARC03700
 BEGIN                                                                         00008560    ARC03700
   READCDATAIN [NO],A,CODE)[PROCESS];                                          00008570    ARC03700
   IF CODE≠"PAC" THEN GO TO REPORTS;                                           00008580    ARC03710
   READCDATAIN,/,I,X1,Y1,X2,Y2);                                               00008590    ARC03710
   FORJ / DO                                                                   00008600    ARC03720
    BEGIN                                                                      00008610    ARC03720
      T[J]←0; U[J]←0                                                           00008620    ARC03720
    END;                                                                       00008630    ARC03720
   NPAC←NPAC+1; N3←4×NPAC; CT ←CU←-1; K←Y2;                                    00008640    ARC03740
   PICKCHARS(X2,K,7,5,2);                                                      00008650    ARC03740
   PICKCHARS(Y1,K,7,3,2); PACDTAN3←K;                                          00008660    ARC03740
   PICKCHARS(X1,PACDTAN3,7,1,2); N3←N3+1;                                      00008670    ARC03750
   KC←SQRTGTCX2,Y2,0);                                                         00008680    ARC03760
   FOR KD←1 STEP 1 UNTIL KC DO                                                 00008690    ARC03760
    BEGIN                                                                      00008700    ARC03760
      I←SQRTGTCX2,Y2,KD); KE←TCITAB[1,1];                                      00008710    ARC03770
      KA←0; C←COVERED(I);                                                      00008720    ARC03770
      FOR K←8 STEP -1 UNTIL 1 DO                                               00008730    ARC03780
       BEGIN                                                                   00008740    ARC03780
         PICKCHARS(KE,KA,K,8,1);                                               00008750    ARC03780
         IF KA=0 THEN GO TO L5;                                                00008760    ARC03790
         IF KA=" " THEN GO TO L4;                                              00008770    ARC03790
         IF CT=NTYPT-1 THEN IF C THEN GO TO L4 ELSE                            00008780    ARC03800
         GO TO L3;                                                             00008790    ARC03800
         FORJ CT DO IF KA=T[J] THEN IF C THEN GO TO                            00008800    ARC03810
         L4 ELSE GO TO L3; CT←CT+1;                                            00008810    ARC03820
         T[CT]←KA;                                                             00008820    ARC03820
         IF NOT C THEN                                                         00008830    ARC03820
          BEGIN                                                                00008840    ARC03820
            CU←CU+1; U[CU]←KA                                                  00008850    ARC03820
          END;                                                                 00008860    ARC03820
         GO TO L4 ;                                                            00008870    ARC03830
L3:      FORJ CU DO IF KA=U[J] THEN GO TO L4;                                  00008880    ARC03840
         CU←CU+1; U[CU]←KA;                                                    00008890    ARC03840
L4:      IF CU=NTYPT-1 THEN GO TO FIN                                          00008900    ARC03850
       END;                                                                    00008910    ARC03850
L5:  END;                                                                      00008920    ARC03850
FIN: IF T[0]=0 THEN                                                            00008930    ARC03860
     BEGIN                                                                     00008940    ARC03860
       NPAC←NPAC-1;                                                            00008950    ARC03860
       WRITECDATAOUT,<"NO TARGETS FOR PAC X=",I4," Y=",                        00008960    ARC03870
       I4, "TO X=",I4," Y=",I4 >,X1,Y1,X2,Y2);                                 00008970    ARC03880
       GO TO FINISH                                                            00008980    ARC03880
     END ELSE                                                                  00008990    ARC03880
     BEGIN                                                                     00009000    ARC03880
       PACKWORD(PACDTAN3,T); N3←N3+1;                                          00009010    ARC03890
       PACKWORD(PACDTAN3,U);                                                   00009020    ARC03890
     END;                                                                      00009030    ARC03890
   C←TRUE; I←PERIOD; PERIOD←0; N3←N3+1;                                        00009040    ARC03900
   GENAMDATA(0,X1+.5,Y1+.5,X2+.5,Y2+.5,89,NPAC,FULL,C)                         00009050    ARC03910
   ; PERIOD←I; AMDATA[1,0]←0; K←1;                                             00009060    ARC03930
L6: KA←AMDATA[K,8];                                                            00009070    ARC03930
   IF KA59 OR NODEFENSE THEN                                                   00009080    ARC03930
    BEGIN                                                                      00009090    ARC03930
      NPAC←NPAC-1; GO TO FINISH                                                00009100    ARC03940
    END;                                                                       00009110    ARC03940
   FOR J←10 STEP 1 UNTIL KA DO                                                 00009120    ARC03950
    BEGIN                                                                      00009130    ARC03950
      NPACOMS←NPACOMS+1;                                                       00009140    ARC03960
      PACOMS[NPACOMS]←AMDATA[K,J];                                             00009150    ARC03960
```

```
                    END;                                                    00009160  ARC03960
                   PACOTAN3←NPACOMS; KB←AMDATA[K,7];                        00009170  ARC03970
                   IF KB≠0 THEN                                             00009180  ARC03970
                    BEGIN                                                   00009190  ARC03970
                      AMDATA[K,0]←0; K←KB; GO TO L6;                        00009200  ARC03970
                    END;                                                    00009210  ARC03970
FINISH:           END;                                                      00009220  ARC03980
                END PLAUSIBLE;                                              00009230  ARC03980
                                                                            00009240  ARC03981
COMMENT SETHEUR SETS HEURISTIC VARIABLES TRUE AS SPECIFIED BY INPUTS;       00009250  ARC03981
PROCEDURE      SETHEUR;                                                     00009260  ARC03990
               BEGIN                                                        00009270  ARC03990
LABEL            L1,L2,L3,L4,L5,L6,L7,L8,L9,L10,FIN;                        00009280  ARC04000
SWITCH           SW←L1,L2,L3,L4,L5,L6,L7,L8,L9,L10;                         00009290  ARC04010
                 READ(DATAIN,/,K,K); GO TO SW[K];                           00009300  ARC04020
L1:              HEURISTIC1← IF K=1 THEN TRUE ELSE FALSE;                   00009310  ARC04030
                 GO TO FIN;                                                 00009320  ARC04030
L2:              HEURISTIC2← IF K=2 THEN TRUE ELSE FALSE;                   00009330  ARC04040
                 GO TO FIN;                                                 00009340  ARC04040
L3:              HEURISTIC3← IF K=3 THEN TRUE ELSE FALSE;                   00009350  ARC04050
                 GO TO FIN;                                                 00009360  ARC04050
L4:              HEURISTIC4← IF K=4 THEN TRUE ELSE FALSE;                   00009370  ARC04060
                 GO TO FIN;                                                 00009380  ARC04060
L5:              HEURISTIC5← IF K=5 THEN TRUE ELSE FALSE;                   00009390  ARC04070
                 GO TO FIN;                                                 00009400  ARC04070
L6:              HEURISTIC6← IF K=6 THEN TRUE ELSE FALSE;                   00009410  ARC04080
                 GO TO FIN;                                                 00009420  ARC04080
L7:              HEURISTIC7← IF K=7 THEN TRUE ELSE FALSE;                   00009430  ARC04090
                 GO TO FIN;                                                 00009440  ARC04090
L8:              HEURISTIC8← IF K=8 THEN TRUE ELSE FALSE;                   00009450  ARC04100
                 GO TO FIN;                                                 00009460  ARC04100
L9:              HEURISTIC9← IF K=9 THEN TRUE ELSE FALSE;                   00009470  ARC04110
                 GO TO FIN;                                                 00009480  ARC04110
L10:             HEURISTIC10←IF K=10 THEN TRUE ELSE FALSE;                  00009490  ARC04120
                 GO TO FIN;                                                 00009500  ARC04120
FIN:             WRITE(DATAOUT, HEUR[K]);                                   00009510  ARC04130
               END SETHEUR;                                                 00009520  ARC04130
                                                                            00009530  ARC04131
COMMENT SELECTDM SELECTS A TDM FOR USE AGAINST THE AM STORED IN ROW ALTX     00009540  ARC04131
        OF TABLE ALTERS. IT MAY CHANGE TDM PREVIOUSLY SELECTED;             00009550  ARC04132
PROCEDURE      SELECTTDM;                                                   00009560  ARC04140
               BEGIN                                                        00009570  ARC04140
INTEGER          I,J,K,KA,KB,KC,KD,KE,KF,L,KG,KH,KI,A,KJ;                  00009580  ARC04150
LABEL            FIN,L1,L2,L3;                                              00009590  ARC04160
                 ITDMONE←ITDMZERO←ITDMONEA←ITDMZEROA←0;                     00009600  ARC04170
                 TDMONE←TDMZERO←FFAS←FALSE;                                 00009610  ARC04170
                 KA←ALTERS[ALTX,4]; KB←ALTERS[ALTX,KA];                     00009620  ARC04180
                 KD←AMDATA[ALTERS[ALTX,0],3];                               00009630  ARC04180
                 IF KB=0 THEN GO TO L2;                                     00009640  ARC04190
                      COMMENT BRANCH TO SELECT NEW TDM FOR ALTX;            00009650  ARC04190
                 KC←TDLIST[KB]←TDLIST[KB]+1;                                00009660  ARC04200
                 ALTERS[ALTX,KA]←0; IF KC≥3 THEN GO TO FIN;                 00009670  ARC04210
                 A←0;                                                       00009680  ARC04210
L1:              FOR I ← 1 DO                                               00009690  ARC04230
                  BEGIN                                                     00009700  ARC04230
                    KC←TDMTGT[KB,I]; IF KC=0 THEN GO TO FIN;                00009710  ARC04230
                    KE←A;                                                   00009720  ARC04230
                    FOR J←2,3,4,5 DO                                        00009730  ARC04240
                     BEGIN                                                  00009740  ARC04240
                       KE←KE+TDLIST[TGTTAD[KC,J]];                          00009750  ARC04240
                       IF KE≥3 THEN GO TO L3;                               00009760  ARC04250
```

```
                    END;                                                    00009770        ARC04250
                   IF KE=1 THEN                                             00009780        ARC04260
                    BEGIN                                                   00009790        ARC04260
                      ITDMZERO←ITDMZERO+1;                                  00009800        ARC04260
                      TDMZERO←TDMONE←TRUE;                                  00009810        ARC04260
                      ITDMONE←ITDMONE-1                                     00009820        ARC04270
                    END;                                                    00009830        ARC04270
                   IF KE=2 THEN                                             00009840        ARC04280
                    BEGIN                                                   00009850        ARC04280
                      ITDMONE←ITDMONE+1; TDMONE←TRUE;                       00009860        ARC04280
                    END;                                                    00009870        ARC04280
                   IF KE<3 THEN IF ACTIVE(TYPATK,TGTTAD[KC,1]) THEN         00009880        ARC04290
                   IF KE=1 THEN                                             00009890        ARC04290
                    BEGIN                                                   00009900        ARC04290
                      ITDMZERDA←ITDMZERDA+1;                                00009910        ARC04300
                      ITDMONEA←ITDMONEA-1                                   00009920        ARC04300
                    END ELSE ITDMONEA← ITDMONEA+1;                          00009930        ARC04310
L3:                FND FORI LOOP;                                           00009940        ARC04320
                   GO TO FIN;                                               00009950        ARC04320
L2:                KE←-1; A←1;                                              00009960        ARC04330
                   FOR J←2,3,4,5 DO                                         00009970        ARC04340
                    BEGIN                                                   00009980        ARC04340
                      I←TGTTAD[KD,J]; KE←TDLIST[I];                         00009990        ARC04340
                      IF KE>KE THEN                                         00010000        ARC04340
                       BEGIN                                                00010010        ARC04340
                         KE←KF; K←I;                                        00010020        ARC04350
                       END;                                                 00010030        ARC04350
                    END;                                                    00010040        ARC04350
                   IF KE>0 THEN                                             00010050        ARC04360
                    BEGIN                                                   00010060        ARC04360
                      TDLIST[K]←TDLIST[K]-1;                                00010070        ARC04360
                      ALTERS[ALTX,KA]←K; FEAS←TRUE; TDMAFF←K;               00010080        ARC04370
                      IF KE>2 THEN GO TO FIN ELSE                           00010090        ARC04360
                       BEGIN                                                00010100        ARC04360
                         KB←K; GO TO L1;                                    00010110        ARC04380
                       END                                                  00010120        ARC04380
                    END;                                                    00010130        ARC04380
                   FORK LASTALT DO IF K≠ALTX THEN                           00010140        ARC04390
                    BEGIN                                                   00010150        ARC04390
                      KJ←ALTERS[K,ALTERS[K,4]];                             00010160        ARC04390
                      IF KJ<@4 AND KJ≠0 THEN                                00010170        ARC04400
                       BEGIN                                                00010180        ARC04400
                         FOR J←2,3,4,5 DO                                   00010190        ARC04410
                          BEGIN                                             00010200        ARC04410
                            I←TGTTAD[KD,J];                                 00010210        ARC04410
                            IF KJ=I THEN                                    00010220        ARC04410
                             BEGIN                                          00010230        ARC04410
                               KG←0; KE←AMDATA[ALTERS[K,0],3];              00010240        ARC04420
                               FOR L←2,3,4,5 DO                             00010250        ARC04420
                                BEGIN                                       00010260        ARC04420
                                  KE←TGTTAD[KE,L];                          00010270        ARC04420
                                  KH←TDLIST[KE];                            00010280        ARC04430
                                  IF KH>KG THEN                             00010290        ARC04430
                                   BEGIN                                    00010300        ARC04430
                                     KG←KH; KI←KE;                          00010310        ARC04430
                                   END;                                     00010320        ARC04430
                                END;                                        00010330        ARC04430
                               IF KG>0 THEN                                 00010340        ARC04440
                                BEGIN                                       00010350        ARC04440
                                  ALTERS[K,ALTERS[K,4]]←KI;                 00010360        ARC04440
                                  ALTERS[ALTX,KA]←I;                        00010370        ARC04440
```

```
                                TDLIST[KI]←TDLIST[KI]-1;                          00010380  ARC04450
                                FEAS←TRUE; TOMAFF←KI;                             00010390  ARC04450
                                IF KG>2 THEN GO TO FIN ELSE                       00010400  ARC04460
                                 BEGIN                                            00010410  ARC04460
                                   KB←KI; GO TO L1                                00010420  ARC04460
                                 END                                              00010430  ARC04460
                               END;                                               00010440  ARC04460
                             END IF ALTERS BLOCK;                                 00010450  ARC04470
                           END FOR J LOOP;                                        00010460  ARC04480
                         END;                                                     00010470  ARC04480
                       END FORK LOOP;                                             00010480  ARC04480
FIN:                 END SELECTTOM;                                               00010490  ARC04490
                                                                                  00010500  ARC04491
COMMENT EVALUATE CALCULATES THE FEATURE VALUES GIVEN THAT THE DM STORED           00010510  ARC04491
        IN ROW ALTX,ELEMENT DMOUT OF TABLE ALTERS IS BEING USED INSTEAD           00010520  ARC04492
        OF ELEMENT DMIN. DMIN=0 OR DMOUT=0 INDICATES THAT A DM IS ONLY            00010530  ARC04493
        BEING USED OR ONLY BEING SAVED,RESPECTIVELY;                              00010540  ARC04494
PROCEDURE       EVALUATE(DMIN,DMOUT);                                             00010550  ARC04510
VALUE           DMIN,DMOUT;                                                       00010560  ARC04520
INTEGER         DMIN,DMOUT;                                                       00010570  ARC04520
                 BEGIN                                                            00010580  ARC04520
       COMMENT EVALUATE FEATURE VALUES, DMIN IS ALTERS INDEX OF DM BEING         00010590  ARC04530
    RETURNED,DMOUT IS SAME FOR ASSIGNED;                                          00010600  ARC04540
LABEL               L1,L2,L3,FIN,L4,L5,L6,L7,L8;                                  00010610  ARC04550
BOOLEAN             VOID;                                                         00010620  ARC04550
REAL                X,Y,Z,V;                                                      00010630  ARC04550
INTEGER             I,J,K,KA,KB,KC,KD,KE,KF,A,B,C,D,N3;                           00010640  ARC04560
                    SUBEVALCNT←SUBEVALCNT+1; B←0;                                 00010650  ARC04570
                    IF DMOUT=0 THEN                                               00010660  ARC04580
                     BEGIN                                                        00010670  ARC04580
                       IF DMIN=0 THEN                                             00010680  ARC04580
                        BEGIN                                                     00010690  ARC04580
                          VOID←TRUE;                                              00010700  ARC04580
                          GO TO L3                                                00010710  ARC04580
                        END                                                       00010720  ARC04580
                     END ELSE                                                     00010730  ARC04590
                     BEGIN                                                        00010740  ARC04590
                       IF BPAC THEN BREAKDMWORD(WPAC,A,B,C,D) ELSE                00010750  ARC04600
                       BREAKDMWORD(ALTERS[ALTX,DMOUT],A,B,C,D);                   00010760  ARC04610
                       IF BPAC OR B≠0 THEN IF CHEKBIT(DMVOID[D],C) THEN           00010770  ARC04620
                        BEGIN                                                     00010780  ARC04620
                          FEAS←FALSE; GO TO FIN                                   00010790  ARC04630
                        END;                                                      00010800  ARC04630
                     END;                                                         00010810  ARC04630
                    FEAS←TRUE; VOID←FALSE; Z←1;                                   00010820  ARC04640
                    V←IF BPAC THEN PACVALUE ELSE TGTLCV[AMDATA[ALTERS[            00010830  ARC04650
                    ALTX,0],3],A]; IF BPAC THEN GO TO L1;                         00010840  ARC04660
                    IF DMIN=0 THEN IF B=0 THEN GO TO L2 ELSE GO TO L1;            00010850  ARC04670
                    BREAKDMWORD(ALTERS[ALTX,DMIN],KA,KB,KC,KD);                   00010860  ARC04680
                    IF KB=0 THEN GO TO L1; KE←DMLOCO[KC,KD+3];                    00010870  ARC04690
                    KF←DMLOCO[KC,2]; KE←KE+1;                                     00010880  ARC04690
                    DMLOCO[KC,KD+3]←KE;                                           00010890  ARC04690
                    IF KE=1 THEN                                                  00010900  ARC04700
                     BEGIN                                                        00010910  ARC04700
                       CLEARBIT(DMVOID[KD],KC); VOID←TRUE;                        00010920  ARC04700
                       TFEAT[17]←TFEAT[17]-1;                                     00010930  ARC04710
L7:                  END;                                                         00010940  ARC04710
                    IF CHEKBIT(TWOCOR,KC) THEN TFEAT[12]←TFEAT[12]-1 ELSE         00010950  ARC04730
                    IF CHEKBIT(THREECOR,KC) THEN TFEAT[13]←TFEAT[13]-1;           00010960  ARC04730
                    X←KE/KF; Y←(KE-1)/KF;                                         00010970  ARC04740
                    IF Y≤.25 THEN                                                 00010980  ARC04750
```

```
                    BEGIN                                                                00010990  ARC04750
                      IF X>.25 THEN                                                      00011000  ARC04750
                       BEGIN                                                             00011010  ARC04750
                         TFEAT[16]←TFEAT[16]-Z;                                          00011020  ARC04750
                         TFEAT[15] ←TFEAT[15]+Z                                          00011030  ARC04760
                       END                                                               00011040  ARC04760
                    END ELSE IF Y≤.5 THEN                                                00011050  ARC04760
                    BEGIN                                                                00011060  ARC04760
                      IF X>.5 THEN                                                       00011070  ARC04760
                       BEGIN                                                             00011080  ARC04760
                         TFEAT[15]←TFEAT[15]-Z;                                          00011090  ARC04770
                         TFEAT[14]←TFEAT[14]+Z                                           00011100  ARC04770
                       END                                                               00011110  ARC04770
                    END ELSE IF Y≤.75 THEN IF X>.75 THEN TFEAT[14]←TFEAT[                00011120  ARC04780
                    14]-Z; IF DMOUT=0 THEN GO TO L3;                                     00011130  ARC04790
                    IF B≠0 AND NOT BPAC THEN GO TO L2;                                   00011140  ARC04800
L1:                 KE←DMLOCO[C,D+3]; KF←DMLOCO[C,2]; KF←KE-1;                           00011150  ARC04810
                    DMLOCO[C,D+3]←KE;                                                    00011160  ARC04810
                    IF KE=0 THEN                                                         00011170  ARC04820
                     BEGIN                                                               00011180  ARC04820
                       SETBIT(DMVOID[D],C); VOID←TRUE;                                   00011190  ARC04820
                       TFEAT[17]←TFEAT[17]+1;                                            00011200  ARC04830
                     END;                                                                00011210  ARC04830
L8:                 IF CHEKBIT(TWOCOR,C) THEN TFEAT[12]←TFEAT[12]+1 ELSE                 00011220  ARC04850
                    IF CHEKBIT(THREECOR,C) THEN TFEAT[13]←TFEAT[13]+1;                   00011230  ARC04850
                    X←KE/KF; Y←(KF+1)/KF;                                                00011240  ARC04860
                    IF X≤.25 THEN                                                        00011250  ARC04870
                     BEGIN                                                               00011260  ARC04870
                       IF Y>.25 THEN                                                     00011270  ARC04870
                        BEGIN                                                            00011280  ARC04870
                          TFEAT[16]←TFEAT[16]+Z;                                         00011290  ARC04870
                          TFEAT[15] ←TFEAT[15]-Z                                         00011300  ARC04880
                        END                                                              00011310  ARC04880
                     END ELSE IF X≤.5 THEN                                               00011320  ARC04880
                     BEGIN                                                               00011330  ARC04880
                       IF Y>.5 THEN                                                      00011340  ARC04880
                        BEGIN                                                            00011350  ARC04880
                          TFEAT[15]←TFEAT[15]+Z;                                         00011360  ARC04890
                          TFEAT[14]←TFEAT[14]-Z                                          00011370  ARC04890
                        END                                                              00011380  ARC04890
                     END ELSE IF X≤.75 THEN IF Y>.75 THEN TFEAT[14]←TFEAT[               00011390  ARC04900
                    14]+Z;                                                               00011400  ARC04900
                    IF DMIN=0 OR KB≠0 THEN IF VOID THEN GO TO L3 ELSE GO                 00011410  ARC04910
                    TO FIN;                                                              00011420  ARC04910
L2:  COMMENT TDM SELECTION HAS BEEN MADE;                                                00011430  ARC04920
                    IF NOT TDMONE THEN GO TO L3; KE←ITDMZERO;                            00011440  ARC04930
                    KF←ITDMONE;                                                          00011450  ARC04930
                    IF B≠0 OR DMOUT=0 THEN                                               00011460  ARC04940
                     BEGIN                                                               00011470  ARC04940
                       KE←-KE; KF←-KF                                                    00011480  ARC04940
                     END;                                                                00011490  ARC04940
                    TFEAT[5]←TFEAT[5]+KE; TFEAT[7]←TFEAT[7]+KF;                          00011500  ARC04950
                    KE←ITDMZEROA; KF←ITDMONEA;                                           00011510  ARC04960
                    IF B≠0 OR DMOUT=0 THEN                                               00011520  ARC04970
                     BEGIN                                                               00011530  ARC04970
                       KE←-KE; KF←-KF                                                    00011540  ARC04970
                     END;                                                                00011550  ARC04970
                    TFEAT[6]←TFEAT[6]+KE; TFEAT[8]←TFEAT[8]+KF;                          00011560  ARC04980
                    IF FIRSTPASS THEN GO TO FIN;                                         00011570  ARC04990
L3:                 IF VOID OR TDMZERO THEN K←K ELSE GO TO FIN ;                         00011580  ARC05000
                    FOR I1 4 DO TFEAT[I]←0;                                              00011590  ARC05010
```

```
                SOLEDM[0]←SOLDMA[0]←0;                                          00011600  ARC05010
                FOR I1 NPAC DO                                                  00011610  ARC05020
                 BEGIN                                                          00011620  ARC05020
                   N3←I×4-1; KE←KF←0; KC←PACDTAN3+1;                            00011630  ARC05020
                   N3←N3+4; KB←PACDTAN3; KD←0;                                  00011640  ARC05030
                   FOR J←KC STEP 1 UNTIL KB DO                                  00011650  ARC05040
                    BEGIN                                                       00011660  ARC05040
                      BREAKDMWORD(PACDMS[J],A,B,C,D);                           00011670  ARC05040
                      IF NOT CHEKBIT(DMVOID[D],C) THEN IF KD=0 THEN             00011680  ARC05050
                      KD←C ELSE KD←-1                                           00011690  ARC05050
                    END;                                                        00011700  ARC05050
                   IF KD<0 THEN GO TO L4;                                       00011710  ARC05060
                   IF KD>0 THEN                                                 00011720  ARC05070
                    BEGIN                                                       00011730  ARC05070
                      N3←N3-2;                                                  00011740  ARC05070
                      IF ACTIVE(TYPATK,PACDTAN3) THEN SETBIT(SOLDMA,            00011750  ARC05080
                      KD) ELSE SETBIT(SOLEDM,KD); N3←N3+2;                      00011760  ARC05090
                      GO TO L4;                                                 00011770  ARC05090
                    END;                                                        00011780  ARC05090
                   TFEAT[1]←TFEAT[1]+1; N3←N3-2;                                00011790  ARC05110
                   IF ACTIVE(TYPATK,PACDTAN3) THEN TFEAT[2]←TFEAT[2]+           00011800  ARC05120
                   1; N3←N3+1;                                                  00011810  ARC05120
                   IF ACTIVE(TYPATK,PACDTAN3) THEN                              00011820  ARC05130
                    BEGIN                                                       00011830  ARC05130
                      KE←KE+1; GO TO L6                                         00011840  ARC05130
                    END;                                                        00011850  ARC05130
                   KA←KB←0; N3←N3-2;                                            00011860  ARC05150
                   PICKCHARS(PACDTAN3,KA,5,7,2);                                00011870  ARC05150
                   PICKCHARS(PACDTAN3,KB,7,7,2);                                00011880  ARC05150
                   KD←SORTGT[KA,KB,0];                                          00011890  ARC05160
                   FOR J1 KD DO                                                 00011900  ARC05160
                    BEGIN                                                       00011910  ARC05160
                      KC←SORTGT[KA,KB,J];                                       00011920  ARC05160
                      FOR K←2,3,4,5 DO IF TDL1ST[TGTTAD[KC,K]]≠0 THEN           00011930  ARC05170
                      GO TO L5; KE←1;                                           00011940  ARC05180
                      IF ACTIVE(TYPATK,TGTTAD[KC,1]) THEN                       00011950  ARC05180
                       BEGIN                                                    00011960  ARC05180
                         KF←1; GO TO L6;                                        00011970  ARC05190
                       END;                                                     00011980  ARC05190
                    END FOR J LOOP;                                             00011990  ARC05190
L5:                 TFEAT[3]←TFEAT[3]+KE;                                       00012000  ARC05200
L6:                 TFEAT[4]←TFEAT[4]+KF;                                       00012010  ARC05200
L4:              END FOR I LOOP;                                                00012020  ARC05210
FIN:             IF FEAS THEN                                                   00012030  ARC05230
                  BEGIN                                                         00012040  ARC05230
                    FOR I←9,10,11 DO TFEAT[I]←0;                                00012050  ARC05240
                    TFEAT[18]←TFEAT[19]←0;                                      00012060  ARC05240
                    FOR I LASTALT DO                                            00012070  ARC05250
                     BEGIN                                                      00012080  ARC05250
                       BREAKDMWORD(ALTERS[I,ALTERS[I,4]],A,B,C,D);              00012090  ARC05250
                       IF CHEKBIT(SOLEDM,C) THEN TFEAT[18]←TFEAT[18]+1          00012100  ARC05260
                       ELSE IF CHEKBIT(SOLDMA,C) THEN TFEAT[19]←TFEAT[         00012110  ARC05270
                       19]+1;                                                   00012120  ARC05270
                       IF B=0 THEN J←0 ELSE J←REMDM[I,B];                       00012130  ARC05280
                       IF J≤2 THEN                                              00012140  ARC05290
                        BEGIN                                                   00012150  ARC05290
                          V←TGTLCV[AMDATA[ALTERS[I,0],3],4];                    00012160  ARC05290
                          IF B=0 AND D=0 THEN X←1 ELSE X←(IF B=0 THEN           00012170  ARC05310
                          1-DMPROB ELSE 1-DMSPEC[2×D+1]);                       00012180  ARC05310
                          X←X×V; TFEAT[9+J]←TFEAT[9+J]+X;                       00012190  ARC05320
                        END;                                                    00012200  ARC05320
```

```
                    END FORI LOOP;                                          00012210  ARC05330
                    ITDMONE←ITDMZERO←ITDMINFA←ITDMZERDA←0;                  00012220  ARC05340
                    IDMONE←TDMZERO←FALSE;                                   00012230  ARC05340
                  END;                                                      00012240  ARC05340
                END EVALUATE;                                               00012250  ARC05360
                                                                            00012260  ARC05361
COMMENT PACEVAL DETERMINES THE FEATURE VALUES GIVEN THAT AN AM ENTERS        00012270  ARC05361
        THE SYSTEM THROUGH A PAC WHICH MAXIMIZES THE MINIMUM DECISION       00012280  ARC05362
        FUNCTION VALUE OVER ALL PAC. RESULT IS STORED IN PFEAT;             00012290  ARC05363
PROCEDURE       PACEVAL;                                                    00012300  ARC05370
                BEGIN                                                       00012310  ARC05370
        COMMENT         DETERMINE MAX-MIN VALUES UNDER ONE PAC ATTACK;      00012320  ARC05380
INTEGER             I,J,K,KA,KB,KC,KD,A,B,C,D,MIN,MAX,L,N3;                 00012330  ARC05390
REAL                VMIN,VMAX,V;                                            00012340  ARC05400
LABEL               Q,L1,L2,L3,L4;                                          00012350  ARC05400
REAL ARRAY          DMFEAT[0:NDM*NTYPE,0:NFEATURES], INTA[0:NDM*NTYPE];     00012360  ARC05420
INTEGER ARRAY       INTB,INTC[0:NDM*NTYPE];                                 00012370  ARC05420
                    FORI NDM DO INTA[I]←0; BPAC←TRUE; MAX←0;                00012380  ARC05430
                    N3←3; FORI NFEATURES DO XFEAT[I]←TFEAT[I];              00012390  ARC05440
                    FORI NFEATURES DO DMFEAT[0,I]←TFEAT[I];                 00012400  ARC05450
                    INTB[0]←TFEAT[4]; INTC[0]←TFEAT[3];                     00012410  ARC05460
                    MULTCWATE,TFEAT,INTA[0],K,NFEATURES);                   00012420  ARC05460
                    FORI1 NPAC DO                                           00012430  ARC05470
                     BEGIN                                                  00012440  ARC05470
                       C←PACDTAN3; N3←N3+4; D←PACDTAN3; MIN←0;              00012450  ARC05480
                       FOR J← C+1 STEP 1 UNTIL D DO                         00012460  ARC05490
                        BEGIN                                               00012470  ARC05490
                          WPAC←PACDMS[J];                                   00012480  ARC05490
                          BREAKDWORD(WPAC ,A,A,A,B);                        00012490  ARC05500
                          IF INTA[A]≠0 THEN GO TO L1;                       00012500  ARC05500
                          FORK NFEATURES DO TFEAT[K]←XFEAT[K];              00012510  ARC05510
                          K←DMLOCDTA,B,3];                                  00012520  ARC05520
                          IF K=0 THEN GO TO Q ; EVALUATE(0,1);              00012530  ARC05520
                          DMLOCDTA,B,3]←K;                                  00012540  ARC05520
                          IF K=1 THEN CLEARSTI(DMVDIDTR],A);                00012550  ARC05530
                          FORK NFEATURES DO DMFEAT[A+B*NDM,K]←TFEAT[K];      00012560  ARC05540
                          INTC[A]←TFEAT[3];                                 00012570  ARC05540
                          MULTCWATE,TFEAT,INTA[A],K,NFEATURES);             00012580  ARC05550
                          INTB[A]←TFEAT[4];                                 00012590  ARC05550
L1:                       V←INTA[A];                                        00012600  ARC05560
                          IF HEURISTIC9 THEN KA←KC←0 ELSE                   00012610  ARC05560
                           BEGIN                                            00012620  ARC05560
                             KA←INTB[A]; KC←INTC[A]                         00012630  ARC05570
                           END;                                             00012640  ARC05570
                          IF MIN=0 THEN GO TO L2;                           00012650  ARC05570
                          VMIN←INTA[MIN];                                   00012660  ARC05570
                          IF HEURISTIC9 THEN KB←KD←0 ELSE                   00012670  ARC05580
                           BEGIN                                            00012680  ARC05580
                             KB←INTB[MIN]; KD←INTC[MIN]                     00012690  ARC05580
                           END;                                             00012700  ARC05580
                          IF KA<KB THEN GO TO L2;                           00012710  ARC05590
                          IF KB<KA OR VMIN<V+B-5 THEN GO TO Q;              00012720  ARC05600
                          IF V<VMIN OR KC<KD THEN GO TO L2 ELSE GO TO Q;    00012730  ARC05610
L2:                       MIN←A;                                            00012740  ARC05620
                          IF MAX=0 THEN IF J=D THEN GO TO L4 ELSE GO TO Q;  00012750  ARC05630
                          VMAX←INTA[MAX];                                   00012760  ARC05640
                          IF HEURISTIC9 THEN KB←KD←0 ELSE                   00012770  ARC05640
                           BEGIN                                            00012780  ARC05640
                             KB←INTB[MAX]; KB←INTC[MAX]                     00012790  ARC05650
                           END;                                             00012800  ARC05650
                          IF KA<KB THEN GO TO L4;                           00012810  ARC05660
```

```
                    IF KR<KA OR VMAX<V+8=5 THEN GO TO Q;                    00012820    ARC05670
                    IF V<VMAX OR KC<KD THEN GO TO L4 ELSE GO TO Q;          00012830    ARC05680
L3;                 MAX←MIN;                                                00012840    ARC05690
Q;                  END FOR JLOOP;                                          00012850    ARC05700
                   MAX←IF MIN=0 THEN MAX ELSE MIN;                          00012860    ARC05700
L4;               END FORI LOOP;                                            00012870    ARC05710
                  FORK NFEATURES DO TFEAT[K]←XFEAT[K];                      00012880    ARC05710
                  TV1←INTR[MAX]; TV2←INTA[MAX];                             00012890    ARC05720
                  TV3←INTC[MAX]; BEAC←FALSE;                                00012900    ARC05720
                  IF HEURISTICS THEN TV1←TV3←0;                             00012910    ARC05730
                  FORK NFEATURES DO PFEAT[K]←DMFEAT[MAX,K];                 00012920    ARC05740
                  PFEAT[0]←TV2;                                             00012930    ARC05740
                END PACEVAL;                                                00012940    ARC05750
                                                                            00012950    ARC05751
COMMENT NEWAM SUPERVISES THE GENERATION OF AMDATA BY EITHER CALLING PROC    00012960    ARC05751
        GENAMDATA OR RETRIEVING DATA PREVIOUSLY STORED IN PAMDATA;          00012970    ARC05752
PROCEDURE       NEWAM(PERIOD,I);                                            00012980    ARC05760
VALUE           PERIOD,I;                                                   00012990    ARC05760
INTEGER         PERIOD,I;                                                   00013000    ARC05760
                 BEGIN                                                      00013010    ARC05760
REAL               Q;                                                       00013020    ARC05760
                   AMIND←AMIND+1; ATTACK[I,7]←1;                            00013030    ARC05770
                   NOTGT←NODEFENSE←TOOLATE←FALSE;                           00013040    ARC05770
                   IF AMFLAG THEN                                           00013050    ARC05780
                    BEGIN                                                   00013060    ARC05780
                      FORJ1 MAMDATA DO IF AMDATA[J,0]=0 THEN                00013070    ARC05790
                       BEGIN                                                00013080    ARC05790
                         LASTPROW←J; J←9999;                                00013090    ARC05790
                       END;                                                 00013100    ARC05790
                      IF MAXAMDT<LASTPROW THEN MAXAMDT←LASTPROW;            00013110    ARC05800
                      FORJ NAMDATA DO AMDATA[LASTPROW,J]←PAMDTA[AMIND,J];   00013120    ARC05810
                      K←AMDATA[LASTPROW,0];                                 00013130    ARC05820
                      IF K≠1 THEN AMDATA[LASTPROW,0]←0;                     00013140    ARC05820
                      IF K≠2 THEN ATKTYPE(AMDATA[LASTPROW,3],PERIOD);       00013150    ARC05830
                      IF K≠1 THEN IF K=2 THEN NODEFENSE←TRUE ELSE IF K=3    00013160    ARC05840
                      THEN NOTGT←TRUE ELSE TOOLATE← TRUE;                   00013170    ARC05840
                      WRITE(DATAOUT,<"AM",I4>,AMDATA[LASTPROW,1]);          00013180    ARC05850
                      IF AMDATA[LASTPROW,7]≠0 THEN                          00013190    ARC05860
                       BEGIN                                                00013200    ARC05860
                         AMIND←AMIND+1;                                     00013210    ARC05860
                         FOR J←LASTPROW STEP 1 UNTIL MAMDATA DO IF          00013220    ARC05870
                         AMDATA[J,0]=0 THEN                                 00013230    ARC05870
                          BEGIN                                             00013240    ARC05870
                            K←J; J←MAMDATA;                                 00013250    ARC05880
                          END;                                              00013260    ARC05880
                         FORJ NAMDATA DO AMDATA[K,J]←PAMDTA[AMIND,J];       00013270    ARC05880
                         IF MAXAMDT<K THEN MAXAMDT←K;                       00013280    ARC05890
                         AMDATA[LASTPROW,7]←K;                              00013290    ARC05900
                         AMDATA[K,0]←-LASTPROW;                             00013300    ARC05900
                       END                                                  00013310    ARC05900
                    END ELSE                                                00013320    ARC05900
                    BEGIN                                                   00013330    ARC05900
                      GENAMDATA(ATTACK[I,1],ATTACK[I,2],ATTACK[I,3],        00013340    ARC05910
                      ATTACK[I,4],ATTACK[I,5], ATTACK[I,6],ATTACK[I,0],     00013350    ARC05920
                      FULL,FALSE);                                          00013360    ARC05920
                      IF NOT NOTGT THEN ATKTYPE(AMDATA[LASTPROW,3],         00013370    ARC05930
                      PERIOD);                                              00013380    ARC05930
                    END IF AMFLAG;                                          00013390    ARC05940
                   IF NODEFENSE THEN                                        00013400    ARC05950
                    BEGIN                                                   00013410    ARC05950
                      K←AMDATA[LASTPROW,3];                                 00013420    ARC05950
```

```
                    KA←AMDATA[LASTPROW,4];                                  00013430  ARC05950
                    T←TGTLCV[K,4]←TGTLCV[K,4]×(1-DESTPROB);                 00013440  ARC05960
                    KB←AMDATA[LASTPROW,1];                                  00013450  ARC05960
                    WRITE(DATAOUT,FUTDEST,PERIOD,KB,KA);                    00013460  ARC05970
                    WRITE(DATAOUT[DBL],DESTTGT,KA,AMDATA[LASTPROW,2],       00013470  ARC05980
                    KB,T); AMNOW←AMNOW-1; ATTACK[I,7]←0;                    00013480  ARC05990
                  END NODEFENSE ELSE IF NOTGT THEN                          00013490  ARC05990
                  BEGIN                                                     00013500  ARC05995
                    WRITE(DATAOUT[DBL],<"AM",I4," HAS NO TARGET">,          00013510  ARC06000
                    ATTACK[I,0]); AMNOW←AMNOW-1;                            00013520  ARC06010
                    ATTACK[I,7]←0                                           00013530  ARC06010
                  END NOTGT ELSE IF TOOLATE THEN                            00013540  ARC06020
                  BEGIN                                                     00013550  ARC06020
                    K←AMDATA[LASTPROW,3]; KA←-1;                            00013560  ARC06020
                    KE←AMDATA[LASTPROW,1];                                  00013570  ARC06020
                    FOR J←2,3,4,5 DO                                        00013580  ARC06030
                     BEGIN                                                  00013590  ARC06030
                       KC←TGTTAD[K,J]; KD←TDLIST[KC];                       00013600  ARC06030
                       IF KA<KD THEN                                        00013610  ARC06030
                        BEGIN                                               00013620  ARC06030
                          KA←KD; KB←KC;                                     00013630  ARC06040
                        END                                                 00013640  ARC06040
                     END;                                                   00013650  ARC06040
                    KF←AMDATA[LASTPROW,4];                                  00013660  ARC06040
                    IF KA≠0 THEN                                            00013670  ARC06050
                     BEGIN                                                  00013680  ARC06050
                       TDLIST[KB]←TDLIST[KB]-1;                             00013690  ARC06050
                       DLIST[KB]←DLIST[KB]-1                                00013700  ARC06050
                     END;                                                   00013710  ARC06050
                    IF KA=0 OR TDMPROB<RANDOM THEN                          00013720  ARC06060
                     BEGIN                                                  00013730  ARC06060
                       T←TGTLCV[K,4]←TGTLCV[K,4]×(1- DESTPROB);             00013740  ARC06070
                       WRITE(DATAOUT [DBL],DESTTGT,KF,AMDATA[LASTPROW,      00013750  ARC06080
                       2],KE,T);                                            00013760  ARC06080
                     END ELSE IF KA≠0 THEN                                  00013770  ARC06090
                     BEGIN                                                  00013780  ARC06090
                       WRITE(DATAOUT[DBL],KILLAM,KE,AMDATA[ LASTPROW,2]     00013790  ARC06100
                       ,"TDM",KB); AMNOW←AMNOW-1                            00013800  ARC06100
                     END;                                                   00013810  ARC06100
                    IF KA≠0 THEN TDLIST[KB]←KA-1;                           00013820  ARC06110
                    ATTACK[I,7]←0                                           00013830  ARC06110
                  END TOOLATE;                                              00013840  ARC06110
                END NEWAM;                                                  00013850  ARC06120
                                                                            00013860  ARC06121
COMMENT GENALTERS GENERATES A SINGLE ROW IN ALTERS FROM AM DATA STORED       00013870  ARC06121
        IN ROW I OF AMDATA;                                                 00013880  ARC06122
PROCEDURE       GENALTERS(I);                                               00013890  ARC06130
VALUE           I;                                                          00013900  ARC06130
INTEGER         I;                                                          00013910  ARC06130
                BEGIN                                                       00013920  ARC06130
LABEL             L1,L2,L3,L4,L5,L, NEXT,LAST;                              00013930  ARC06140
                  KA←FINAL;                                                 00013940  ARC06140
                  IF HEUR2(I,PERIOD) THEN                                   00013950  ARC06150
                   BEGIN                                                    00013960  ARC06150
                     KA←KA+1; ALTERS[KA,0]←I;                               00013970  ARC06150
                     ALTERS[KA,1]← ALTERS[KA,2]←ALTERS[KA,3]←ALTERS[KA,     00013980  ARC06160
                     4]←5; ALTERS[KA,5]←0; GO TO NEXT;                      00013990  ARC06160
                   END;                                                     00014000  ARC06160
                  KB←AMDATA[I,8]; KC←-PERIOD; KF←4;                         00014010  ARC06180
                  IF COVERED(AMDATA[I,3]) THEN BPAC←TRUE ELSE BPAC←         00014020  ARC06190
                  FALSE;                                                    00014030  ARC06190
```

```
                FOR K←AMDATA[I,5] STEP 1 UNTIL KB DO IF AMDATA[I,K]≤   00014040   ARC06200
                KC THEN GO TO L1;                                      00014050   ARC06200
                IF AMDATA[I,7]≠0 THEN                                  00014060   ARC06210
                 BEGIN                                                 00014070   ARC06210
                   AMDATA[AMDATA[I,7],0]←1;                            00014080   ARC06210
                   AMDATA[I,0]←0                                       00014090   ARC06210
                 END ELSE                                              00014100   ARC06220
                 BEGIN                                                 00014110   ARC06220
                   IF RPAC THEN                                        00014120   ARC06220
                    BEGIN                                              00014130   ARC06220
                      KA←KA+1; ALTERS[KA,0]←I;                         00014140   ARC06220
                      ALTERS[KA,5]←0;                                  00014150   ARC06230
                      FORJ1 4 DO ALTERS[KA,J]←5;                       00014160   ARC06230
                      IF AMDATA[I,2]≠PERIOD THEN ALTERS[KA,1]←0;       00014170   ARC06240
                    END;                                               00014180   ARC06240
                   AMDATA[I,5]←KB+1;                                   00014190   ARC06240
                 END;                                                  00014200   ARC06240
                GO TO NEXT;                                            00014210   ARC06250
L1:             KA←KA+1; ALTERS[KA,0]←I; AMDATA[I,5]←K;                00014220   ARC06260
                ALTERS[KA,4]←5; ALTERS[KA,1]←0;                        00014230   ARC06260
                IF AMDATA[I,K]=KC THEN                                 00014240   ARC06270
                 BEGIN                                                 00014250   ARC06270
                          COMMENT SET UP CURRENT PERIOD;               00014260   ARC06270
                   L2:FOR J←K+1 STEP 1 UNTIL KB DO                     00014270   ARC06280
                    BEGIN                                              00014280   ARC06280
                      KG←AMDATA[I,J];                                  00014290   ARC06280
                      IF KG<0 THEN GO TO L4;                           00014300   ARC06290
                      BREAKDMWORD(KG,KD,KD,KD,KE);                     00014310   ARC06290
                      IF CHEKBIT(DMVOID[KE],KD) THEN GO TO L3;         00014320   ARC06300
                      KF←KF+1; ALTERS[KA,KF]←KG;                       00014330   ARC06310
L3:                  END FOR J LOOP;                                   00014340   ARC06320
                   IF KB=NAMDATA THEN                                  00014350   ARC06330
                    BEGIN                                              00014360   ARC06330
                      K←8; I←AMDATA[I,7]; GO TO L2;                    00014370   ARC06330
                    END;                                               00014380   ARC06330
L4:                ALTERS[KA,1]←IF KF=4 THEN 0 ELSE KF;                00014390   ARC06340
                   K←J;                                                00014400   ARC06340
                 END CURRENT PERIOD;                                   00014410   ARC06340
                ALTERS[KA,2]←0;                                        00014420   ARC06350
L5:             FOR J←K+1 STEP 1 UNTIL KB DO                           00014430   ARC06360
                 BEGIN                                                 00014440   ARC06360
                   KG←AMDATA[I,J];                                     00014450   ARC06360
                   IF KG<0 THEN IF HEUR2(I,PERIOD) THEN GO TO LAST     00014460   ARC06370
                   ELSE GO TO L;                                       00014470   ARC06370
                   BREAKDMWORD(KG,KD,KD,KD,KE);                        00014480   ARC06380
                   IF CHEKBIT(DMVOID[KE],KD) THEN GO TO L;             00014490   ARC06380
                   KD←OCCUR(ALTERS[KA,*],KF+1,KG,6,6,3);               00014500   ARC06390
                   IF KD≤KF THEN                                       00014510   ARC06390
                    BEGIN                                              00014520   ARC06390
                      ALTERS[KA,KD]← ALTERS[KA,KD]+ALTERSONE;          00014530   ARC06400
                      GO TO L                                          00014540   ARC06400
                    END;                                               00014550   ARC06400
                   KF←KF+1; ALTERS[KA,KF]←KG;                          00014560   ARC06410
L:                FOR J LOOP;                                          00014570   ARC06420
                IF AMDATA[I,7]≠0 THEN                                  00014580   ARC06430
                 BEGIN                                                 00014590   ARC06430
                   I←AMDATA[I,7]; KB←AMDATA[I,8]; K←8;                 00014600   ARC06430
                   GO TO L5;                                           00014610   ARC06430
                 END;                                                  00014620   ARC06430
                I←ALTERS[KA,0];                                        00014630   ARC06440
LAST:           IF KF=4 AND NOT RPAC THEN                              00014640   ARC06450
```

```
                    BEGIN                                                    00014650        ARC06450
                      KA←KA-1; GO TO NEXT                                    00014660        ARC06450
                    END;                                                     00014670        ARC06450
                   KF←KF+1; ALTERS[KA,KF]←0;                                 00014680        ARC06460
                   IF AMDATA[I,2]=PERIOD THEN ALTERS[KA,1]←KF;               00014690        ARC06460
                   ALTERS[KA,3]←KF; ALTERS[KA,2]←HEUR3(KA);                  00014700        ARC06470
                   KD←ALTERS[KA,2];                                          00014710        ARC06480
                   IF KD=KF AND NOT BPAC THEN ALTERS[KA,2]←KD-1;             00014720        ARC06480
                   CW←CW×(ALTERS[KA,2]-4); CWO←CWO×(KF-4);                   00014730        ARC06490
NEXT:              FINALT←KA;                                                00014740        ARC06500
                 END GENALTERS;                                              00014750        ARC06500
                                                                             00014760        ARC06501
COMMENT INITIALALT ESTABLISHES AN INITIAL ALLOCATION FOR THE CURRENT          00014770        ARC06501
        PERIOD, X IS THE LOWER BOUND INDEX IN ALTERS, Y THE UPPER AND H       00014780        ARC06502
        IS A BOOLEAN WHEN TRUE IT IMPLIES MANDATORY ENGAGEMENT OFALL AM;      00014790        ARC06503
PROCEDURE       INITIALALT(X,Y,H);                                           00014800        ARC06510
VALUE           X,Y,H;                                                       00014810        ARC06520
INTEGER         X,Y;                                                         00014820        ARC06520
BOOLEAN         H;                                                           00014830        ARC06520
                 BEGIN                                                       00014840        ARC06520
      COMMENT DETERMINE INITIAL FEASIBLE ALTERNATIVE, X IS LOWER INDEX       00014850        ARC06530
    AND Y IS UPPER INDEX OF ALTERS, H TRUE IMPLIES MANDATORY ENGAGEMENT;     00014860        ARC06540
BOOLEAN           BINIT;                                                     00014870        ARC06550
INTEGER           A,L,K,I,KA,KB,KC,KD;                                       00014880        ARC06550
LABEL             L1,L2,L3,L4,LA,FIN;                                        00014890        ARC06550
                  FIRSTPASS←TRUE; A←ALTX; L←LASTALT;                         00014900        ARC06560
                  LASTPROW←-1;                                               00014910        ARC06560
                  FOR ALTX←X STEP 1 UNTIL Y DO                               00014920        ARC06570
                   BEGIN                                                     00014930        ARC06570
                     KA←ALTERS[ALTX,4]-1;                                    00014940        ARC06570
                     KC←ALTERS[ALTX,IF H THEN 2 ELSE 3]; K←0;                00014950        ARC06580
                     BINIT←TRUE;                                             00014960        ARC06590
L1:                  KA←KA+1; KB←ALTERS[ALTX,KA];                            00014970        ARC06600
                     IF KB=0 THEN                                            00014980        ARC06600
                      BEGIN                                                  00014990        ARC06600
                        LASTALT←ALTX; KD←ALTERS[ALTX,4];                     00015000        ARC06610
                        ALTERS[ALTX,4]←KA; SELECTTOM;                        00015010        ARC06610
                        ALTERS[ALTX,4]←KD;                                   00015020        ARC06610
                        IF NOT FEAS THEN IF H THEN GO TO L2 ELSE             00015030        ARC06620
                         BEGIN                                               00015040        ARC06620
                           T←TGTLCV[AMDATA[ALTERS[ALTX,0],3],4];             00015050        ARC06630
                           TFEAT[9]←TFEAT[9]+T;                              00015060        ARC06640
                           GO TO LA                                          00015070        ARC06640
                         END                                                 00015080        ARC06640
                      END;                                                   00015090        ARC06640
                     EVALUATE(K,KA); IF FEAS THEN GO TO LA;                  00015100        ARC06650
L2:                  IF KA<KC THEN GO TO L1;                                 00015110        ARC06660
                     IF NOT BINIT THEN GO TO L4;                             00015120        ARC06660
L3:                  BINIT←FALSE; ALTX←ALTX-1;                               00015130        ARC06670
                     IF ALTX<X THEN                                          00015140        ARC06670
                      BEGIN                                                  00015150        ARC06670
                        FEAS←FALSE; GO TO FIN;                               00015160        ARC06670
                      END;                                                   00015170        ARC06670
                     K←ALTERS[ALTX,4];                                       00015180        ARC06680
                     KC←ALTERS[ALTX,IF H THEN 2 ELSE 3];                     00015190        ARC06680
                     IF K≠KC THEN                                            00015200        ARC06690
                      BEGIN                                                  00015210        ARC06690
                        KA←K; GO TO L1                                       00015220        ARC06690
                      END;                                                   00015230        ARC06690
L4:                  KB←ALTERS[ALTX,K];                                      00015240        ARC06700
                     IF KB=0 THEN                                            00015250        ARC06710
```

```
                    BEGIN                                                    00015260  ARC06710
                      T←TGTLCVTAMDATA[ALTERS[ALTX,0],3],4];                  00015270  ARC06710
                      TFEAT[9]←TFEAT[9]+T                                    00015280  ARC06720
                    END ELSE                                                 00015290  ARC06720
                    BEGIN                                                    00015300  ARC06720
                      IF KA<@4 THEN SELECTTOM;                               00015310  ARC06720
                      EVALUATE(K,0)                                          00015320  ARC06730
                    END;                                                     00015330  ARC06730
                   ALTERS[ALTX,4]←5; GO TO L3;                               00015340  ARC06730
LA:                 ALTERS[ALTX,4]←KA;                                       00015350  ARC06740
                  END FOR ALTX LOOP;                                         00015360  ARC06750
                 FEAS←TRUE; FIRSTPASS←FALSE; LASTALT←L;                      00015370  ARC06760
                 EVALUATE(0,0);                                              00015380  ARC06760
FIN:             ALTX←A; LASTALT←L;                                          00015390  ARC06770
                END INITIALALT;                                              00015400  ARC06770
                                                                             00015410  ARC06771
COMMENT LOOKAHEAD DETERMINES THE BEST ALLOCATION FOR THE CURRENT PERIOD      00015420  ARC06771
        GIVEN THAT THE NEXT PERIOD AM ARE KNOWN;                             00015430  ARC06772
PROCEDURE      LOOKAHEAD;                                                    00015440  ARC06780
                BEGIN                                                        00015450  ARC06780
       COMMENT CALCULATE DECISION GIVEN NEXT PERIOD AMS;                     00015460  ARC06790
INTEGER           I,J,K,KA,KB,KC,KD,KE,KF,A,L,VA,VC,VD,VE,VF;                00015470  ARC06800
REAL              VB;                                                        00015480  ARC06800
LABEL             L1,L2,L3, L5,LFEAS,FIN;                                    00015490  ARC06810
BOOLEAN           FLIP;                                                      00015500  ARC06810
DEFINE            CHANGE=                                                    00015510  ARC06830
                  BEGIN                                                      00015520  ARC06830
                    LV1←VA;                                                  00015530  ARC06830
                    LV3←VC; TFEAT[0]←LV2←VB;                                 00015540  ARC06830
                    FORI NFEATURES DO                                        00015550  ARC06830
                     BEGIN                                                   00015560  ARC06830
                       LFEATURE[I]←TFEAT[I];                                 00015570  ARC06840
                       LPFEATURE[I]←PFEAT[I]                                 00015580  ARC06840
                     END;                                                    00015590  ARC06840
                    FORI FINALT DO LBESTCOMB[I]←ALTERS[I,ALTERS[I,4]];       00015600  ARC06850
                    IF OUT3 THEN                                             00015610  ARC06860
                     BEGIN                                                   00015620  ARC06860
                       WRITE(DATAOUT[DBL]);                                  00015630  ARC06860
                       WRITE(DATAOUT,<"LBESTCOMB", 30(X1,A2)>,               00015640  ARC06870
                       FORI FINALT DO LBESTCOMB[I]);                         00015650  ARC06870
                       WRITE(DATAOUT[DBL],<10 F12.4>,FORI NFEATURES DO       00015660  ARC06880
                       PFEAT[I])                                             00015670  ARC06880
                     END;                                                    00015680  ARC06880
                  END#,                                                      00015690  ARC06890
                 SETUP=                                                      00015700  ARC06900
                  BEGIN                                                      00015710  ARC06900
                    MULT(WATE,TFEAT,VB,K,NFEATURES);                         00015720  ARC06900
                    IF HEURISTIC9 THEN VA←VC←0 ELSE                          00015730  ARC06910
                     BEGIN                                                   00015740  ARC06910
                       VA←TFEAT[4];                                          00015750  ARC06910
                       VC←TFEAT[3]                                           00015760  ARC06910
                     END                                                     00015770  ARC06910
                  END#,                                                      00015780  ARC06910
                 CK=IF VA<LV1 THEN CHANGE ELSE IF LV1<VA OR LV2<VB+@-5       00015790  ARC06920
                 THEN K←K ELSE IF VB<LV2 OR VC<LV3 THEN CHANGE ELSE IF       00015800  ARC06930
                 VC=LV3 THEN                                                 00015810  ARC06930
                  BEGIN                                                      00015820  ARC06930
                    VD←PFEAT[4]; VE←PFEAT[0]; VF←PFEAT[3];                   00015830  ARC06940
                    IF VD<LPFEATURE[4] THEN CHANGE ELSE IF LPFEATURE[4]      00015840  ARC06950
                    <VD OR LPFEATURE[0]<VE+@-5 THEN K←K ELSE IF VE<          00015850  ARC06960
                    LPFEATURE[0] OR VF<LPFEATURE[3] THEN CHANGE              00015860  ARC06970
```

```
          END#,                                                          00015870   ARC06970
          TVALUE= TGTLCV[AMDATA[ALTERS[ALTX,0],3],4] #;                  00015880   ARC06980
          FOR[1 NFEATURES DO XFEAT[I]←TFEAT[I];                          00015890   ARC06990
          A←ALTX; L←LASTALT; FLAG←FALSE;                                 00015900   ARC06990
          LASTALT←FINALT;                                                00015910   ARC07000
          INITIALALT(NEXTALT,FINALT,TRUE);                               00015920   ARC07010
          IF NOT FEAS THEN INITIALALT(NEXTALT,FINALT,FALSE);             00015930   ARC07020
          SETUP; IF LV2=0 THEN CHANGE ELSE CK;                           00015940   ARC07030
          ALTX←NEXTALT; LASTALT←FINALT;                                  00015950   ARC07040
L2:       IF ALTX>FINALT THEN GO TO FIN;                                 00015960   ARC07050
          KC←ALTERS[ALTX,2]; KA←ALTERS[ALTX,4];                          00015970   ARC07060
          IF KC<KA THEN KC←KA; FLIP←FALSE;                               00015980   ARC07060
          IF KC=5 THEN                                                   00015990   ARC07070
           BEGIN                                                         00016000   ARC07070
             ALTX←ALTX+1; GO TO L2;                                      00016010   ARC07070
           END;                                                          00016020   ARC07070
          IF KA<KC THEN KB←KA+1 ELSE                                     00016030   ARC07080
           BEGIN                                                         00016040   ARC07080
             KB←5; FLIP←TRUE                                             00016050   ARC07080
           END;                                                          00016060   ARC07080
          KE←ALTERS[ALTX,KA];                                            00016070   ARC07090
L1:       KD←ALTERS[ALTX,KB];                                            00016080   ARC07090
          IF KD=0 THEN                                                   00016090   ARC07100
           BEGIN                                                         00016100   ARC07100
             ALTERS[ALTX,4]←KB; SELECTTDM;                               00016110   ARC07100
             ALTERS[ALTX,4]←KA;                                          00016120   ARC07100
             IF NOT FEAS THEN GO TO LFEAS                                00016130   ARC07110
           END ELSE                                                      00016140   ARC07110
           BEGIN                                                        00016150   ARC07110
             BREAKDMWORD(KD,KF,KF,KF, KG);                              00016160   ARC07120
             IF CHEKBIT(DMVOID[KG],KF) THEN GO TO LFEAS                 00016170   ARC07120
           END;                                                         00016180   ARC07120
          IF KE=0 THEN                                                  00016190   ARC07130
           BEGIN                                                        00016200   ARC07130
             FLAG←TRUE; GO TO L3                                        00016210   ARC07130
           END;                                                         00016220   ARC07130
          IF KE<@4 THEN                                                 00016230   ARC07140
           BEGIN                                                        00016240   ARC07140
             SELECTTDM; GO TO L3                                        00016250   ARC07140
           END;                                                         00016260   ARC07140
L3:       ALTERS[ALTX,4]←KB; EVALUATE(KA,KB);                          00016270   ARC07150
          IF FLAG THEN                                                  00016280   ARC07160
           BEGIN                                                        00016290   ARC07160
             FLAG←FALSE;                                                00016300   ARC07160
             TFEAT[9]←TFEAT[9]-TVALUE                                   00016310   ARC07170
           END;                                                         00016320   ARC07170
          SETUP; CK;                                                    00016330   ARC07180
L5:       IF FLIP THEN ALTX←ALTX+1 ELSE ALTX←NEXTALT;                   00016340   ARC07190
          GO TO L2;                                                     00016350   ARC07190
LFEAS:    KB←IF KB<KC THEN KB+1 ELSE 5;                                 00016360   ARC07200
          IF KB=KA THEN                                                 00016370   ARC07210
           BEGIN                                                        00016380   ARC07210
             ALTX←ALTX+1; GO TO L2                                      00016390   ARC07210
           END;                                                         00016400   ARC07210
          IF KB=5 THEN FLIP←TRUE; GO TO L1;                             00016410   ARC07220
FIN:      FOR I←NEXTALT STEP 1 UNTIL FINALT DO                          00016420   ARC07230
           BEGIN                                                        00016430   ARC07230
             KE←ALTERS[I,4]; KF←ALTERS[I,KE];                           00016440   ARC07240
             BREAKDMWORD(KF,KA,KA,KB,KC);                               00016450   ARC07240
             IF KF≠0 THEN IF KA=0 THEN                                  00016460   ARC07250
              BEGIN                                                     00016470   ARC07250
```

```
                    ALTERS[I,KE]←0;                                          00016480    ARC07250
                    TDLIST[KC]←TDLIST[KC]+1                                  00016490    ARC07250
                  END ELSE                                                   00016500    ARC07250
                  BEGIN                                                      00016510    ARC07250
                    KD←DMLOCO[KB,KC+3]←DMLOCO[KB,KC+3]+1;                    00016520    ARC07260
                    IF KD=1 THEN CLEARBIT(DMVOID[KC],KB);                    00016530    ARC07270
                  END;                                                       00016540    ARC07270
                 ALTERS[I,4]←5;                                              00016550    ARC07270
               END FOR I;                                                    00016560    ARC07270
              FORI1 NFEATURES DO TFEAT[I]←XFEAT[I];                          00016570    ARC07280
              ALTX←A; LASTALT←L;                                             00016580    ARC07280
            END LOOKAHEAD;                                                   00016590    ARC07290
                                                                             00016600    ARC07291
COMMENT SETATTACK PREGENERATES AMDATA INFO AND STORES IT IN PAMDATA FOR      00016610    ARC07291
        AN ENTIRE ATTACK, USED DURING TRAINING AND COMPARISON RUNS;          00016620    ARC07292
PROCEDURE      SETATTACK;                                                    00016630    ARC07300
               BEGIN                                                         00016640    ARC07300
       COMMENT SET UP AM DEFENSES IN PAMDATA FOR ENTIRE ATTACK;              00016650    ARC07310
INTEGER          L,I,J,K;                                                    00016660    ARC07320
                 AMIND←0; CURATT←1;                                          00016670    ARC07320
                 FORI1 999 DO                                                00016680    ARC07330
                  BEGIN                                                      00016690    ARC07330
                    PERIOD←I;                                                00016700    ARC07330
                    FOR L←CURATT STEP 1 UNTIL NAM DO IF ATTACK[L,1]≤         00016710    ARC07340
                    PERIOD THEN                                              00016720    ARC07340
                     BEGIN                                                   00016730    ARC07340
                       AMIND←AMIND+1;                                        00016740    ARC07350
                       GENAMDATA(ATTACK[L,1],ATTACK[L,2],ATTACK[L,3],        00016750    ARC07360
                       ATTACK[L,4],ATTACK[L,5], ATTACK[L,6],ATTACK[L,0]      00016760    ARC07370
                       ,FULL,FALSE);                                         00016770    ARC07370
                       FORJ NAMDATA DO PAMDTA[AMIND,J]←AMDATA[LASTPROW,      00016780    ARC07380
                       J];                                                   00016790    ARC07380
                       PAMDTA[AMIND,0]←IF NODEFENSE THEN 2 ELSE IF           00016800    ARC07390
                       NOTGT THEN 3 ELSE IF TOOLATE THEN 4 ELSE 1;           00016810    ARC07400
                       K←AMDATA[LASTPROW,7];                                 00016820    ARC07410
                       IF K≠0 THEN                                           00016830    ARC07410
                        BEGIN                                                00016840    ARC07410
                          AMIND←AMIND+1;                                     00016850    ARC07410
                          FORJ NAMDATA DO PAMDTA[AMIND,J]←AMDATA[K,J];       00016860    ARC07420
                          AMDATA[K,0]←0                                      00016870    ARC07420
                        END;                                                 00016880    ARC07420
                       AMDATA[1,0]←0;                                        00016890    ARC07430
                     END ELSE                                                00016900    ARC07430
                     BEGIN                                                   00016910    ARC07430
                       CURATT←L; L←999;                                      00016920    ARC07430
                     END;                                                    00016930    ARC07430
                    IF L=NAM+1 THEN I←1000                                   00016940    ARC07440
                  END FORI;                                                  00016950    ARC07440
                 PERIOD←MAXAMDT←0; AMFLAG←TRUE; CURATT←1;                    00016960    ARC07450
                 AMIND←0;                                                    00016970    ARC07450
               END SETATTACK;                                                00016980    ARC07460
                                                                             00016990    ARC07470
COMMENT START OF MAIN PROGRAM.                                               00017000    ARC07470
        VARIABLES ARE INITIALIZED, ENVIRONMENT DATA IS READ IN AND STOR-     00017010    ARC07471
        ED, PARAMETERS CONTROLLING THE RUN ARE SET AND THE FIRST ATTACK      00017020    ARC07472
        IS ENTERED. THIS WORK IS DONE FROM HERE TO LABEL NEXTPERIOD;         00017030    ARC07473
              TME←TIME(1);                                                   00017040    ARC07490
              HEURISTIC2←HEURISTIC3←HEURISTIC4←HEURISTIC7←SOOK←              00017050    ARC07500
              HEURISTIC8←FALSE;                                              00017060    ARC07500
              REPEAT←TREPEAT←CYCLE←SAVECYCLE←NUMGRPS ←GRPCUTOFF←0;           00017070    ARC07501
              PACIN←FALSE; REWIND(PACDTAF); REWIND(PACDMSF);                 00017080    ARC07510
```

```
                REBUILDPAC←FALSE;                                                    00017090    ARC07510
                ADJWATE←NEWWATE←LEARNING←LSAVE←BOOK1←BOOK2←HEURISTIC1←               00017100    ARC07520
                HEURISTIC5←FALSE;                                                    00017110    ARC07520
                HEURISTIC6←HEURISTIC9←HEURISTIC10←OUT1←OUT2←OUT3←OUT7←               00017120    ARC07521
                FALSE; ANSWER←ANSCOMB←ANSWAS←RSEED←BPAC←FALSE;                       00017130    ARC07522
                MINATT←1; EVALCNT←SUBEVALCNT←0; CURATT←1;                            00017140    ARC07530
                MAXAMDT←0; MAXPERIOD←60; NPACDMS←0; N3←3;                            00017150    ARC07540
                PACDTAN3←TOTVALUE←AMIND←0; AMFLAG←FALSE;                             00017160    ARC07540
                FORK NTYPE=1 DO SETONES(DUMVOID[K]);                                 00017170    ARC07550
                SETBIT(ALTERSONE,17);                                                00017180    ARC07550
                PERIOD← NDM← NAM←NPAC←NPACDTA←AMDET←AMNOW←0;                         00017190    ARC07560
                ACTIVEINDEX←99;                                                      00017200    ARC07560
                READ(DATAIN,/,MSOR,NSOR,NTYPT,ESTOTAM,TOMPROB,DESTPROB,              00017210    ARC07570
                MINTGTVAL);                                                          00017220    ARC07570
REPORTS:        READ(DATAIN [NO],A,CODE)[PROCESS];                                   00017230    ARC07580
                IF CODE="T  " THEN GENTARGETS;                                       00017240    ARC07590
                IF CODE="SP " THEN GENDMSPEC;                                        00017250    ARC07600
                IF CODE="TDM" THEN GENTDLIST;                                        00017260    ARC07610
                IF CODE="DM " THEN GENDMLOCS;                                        00017270    ARC07620
                IF CODE="AM " THEN GENATTACK;                                        00017280    ARC07630
                IF CODE="GRU" THEN                                                   00017290    ARC07640
                 BEGIN                                                               00017300    ARC07640
                   READ(DATAIN,/,I,I,I,NUMGRPS,K,K,FORJ1 K DO INTC[J]);              00017310    ARC07650
               COMMENT "GROUP,I,OF,NUMGRPS, K,SITES, 1,4,ETC";                       00017320    ARC07650
                   FORJ1 K DO GROUPS[INTC[J]]←I;                                     00017330    ARC07660
                   GO TO REPORTS                                                     00017340    ARC07660
                 END;                                                                00017350    ARC07660
                IF CODE="PAC" THEN PLAUSIBLE;                                        00017360    ARC07670
                IF CODE="HEU" THEN                                                   00017370    ARC07680
                 BEGIN                                                               00017380    ARC07680
                   SETHEUR; GO TO REPORTS                                            00017390    ARC07680
                 END;                                                                00017400    ARC07680
                IF CODE="END" THEN                                                   00017410    ARC07690
                 BEGIN                                                               00017420    ARC07690
                   READ(DATAIN); GO TO PROCESS                                       00017430    ARC07690
                 END;                                                                00017440    ARC07690
                IF CODE="WAT" THEN                                                   00017450    ARC07700
                 BEGIN                                                               00017460    ARC07700
                   READ(DATAIN,/,I,FORI1 NFEATURES DO WATE[I]);                      00017470    ARC07700
                   GO TO REPORTS                                                     00017480    ARC07710
                 END;                                                                00017490    ARC07710
                IF CODE="CUR" THEN                                                   00017500    ARC07720
                 BEGIN                                                               00017510    ARC07720
                   READ(DATAIN,/,I,FORI1 NFEATURES DO REPWATE[I]);                   00017520    ARC07720
                   GO TO REPORTS                                                     00017530    ARC07730
                 END;                                                                00017540    ARC07730
                IF CODE="RAN" THEN                                                   00017550    ARC07740
                 BEGIN                                                               00017560    ARC07740
                   READ(DATAIN,/,I,RSEED); GO TO REPORTS                             00017570    ARC07740
                 END;                                                                00017580    ARC07740
                IF CODE="CUM" THEN                                                   00017590    ARC07751
                 BEGIN                                                               00017600    ARC07751
                   READ(DATAIN); COMPARUN←TRUE;                                      00017610    ARC07751
                   GO REPORTS                                                        00017620    ARC07751
                 END;                                                                00017630    ARC07751
                IF CODE="TRA" THEN                                                   00017640    ARC07750
                 BEGIN                                                               00017650    ARC07750
                   READ(DATAIN,/,I,I,CYCLE,I,REPEAT);                                00017660    ARC07750
 COMMENT "TRAINING,CYCLE,2,REPEAT,10";                                               00017670    ARC07760
                   SAVECYCLE←CYCLE;                                                  00017680    ARC07760
                   IF REPEAT≤10 THEN LSAVE←LEARNING←TRUE;                            00017690    ARC07770
```

```
                GO TO REPORTS;                                                  00017700  ARC07770
              END;                                                              00017710  ARC07770
             IF CODE="ADD" THEN                                                 00017720  ARC07780
              BEGIN                                                             00017730  ARC07780
                READ(DATAIN); PDMX←TRUE;                                        00017740  ARC07780
                GO TO REPORTS                                                   00017750  ARC07790
              END;                                                              00017760  ARC07790
             IF CODE="REM" THEN                                                 00017770  ARC07800
              BEGIN                                                             00017780  ARC07800
                REBUILDPAC←TRUE; READ(DATAIN);                                  00017790  ARC07800
                GO TO REPORTS                                                   00017800  ARC07800
              END;                                                              00017810  ARC07800
             IF CODE="RUN" THEN                                                 00017820  ARC07810
              BEGIN                                                             00017830  ARC07810
                READ(DATAIN,<X7,S A5>,FOR1 4 DO RUNID[I]);                      00017840  ARC07810
                WRITE(DATAOUT,<S A5>,FOR1 4 DO RUNID[I]);                       00017850  ARC07820
                GO TO REPORTS                                                   00017860  ARC07820
              END;                                                              00017870  ARC07820
             IF CODE="ADJ" THEN                                                 00017880  ARC07830
              BEGIN                                                             00017890  ARC07830
                ADJMADE←TRUE; READ(DATAIN);                                     00017900  ARC07830
                GO TO REPORTS                                                   00017910  ARC07830
              END;                                                              00017920  ARC07830
             WRITE(DATAOUT,<"ILLEGAL CARD CODE ",A3>,CODE);                     00017930  ARC07840
             FOR1 R40 DO                                                        00017940  ARC07841
              BEGIN                                                             00017950  ARC07841
                READ(DATAIN,A,CODE)[ENDOFRUN];                                  00017960  ARC07841
                IF CODE="LAS" THEN                                              00017970  ARC07842
                 BEGIN                                                          00017980  ARC07842
                   READ(DATAIN[01:2:1])[ENDOFRUN];                              00017990  ARC07842
                   GO TO STARTCMPPROBLEM                                        00018000  ARC07843
                 END                                                            00018010  ARC07843
              END;                                                              00018020  ARC07843
PROCESS:      GENTDMTGT; IF NOT SOOK THEN GENSQRDTA;                            00018030  ARC07850
              BEGIN                                                             00018040  ARC07850
      COMMENT THIS PSEUDO BLOCK SETS UP TABLES;                                 00018050  ARC07860
INTEGER         I;                                                              00018060  ARC07860
              FOR1 MDMLOCO DO INTA[I]←0; N3←3;                                  00018070  ARC07870
              FOR1 NTYPE←1 DO PICSCHARSCONVDIDE[I].SDMV[I1,1,1,8];              00018080  ARC07880
              FOR1 NTGLIST DO OLTSTE[I]←TOLISTE[I]; K←1;                        00018090  ARC07890
              FOR1 NDM DO FOR J←3 STEP 1 UNTIL NTYPE+2 DO SDLOCDE[K←           00018100  ARC07900
              K+1]←DMLOCDE[,J];                                                 00018110  ARC07900
 COMMENT SAVE TGT VALUE, DM AND TDM  INITIAL MISSILE STOCK;                     00018120  ARC07910
              FOR1 NTGT DO SAVEINE[I]←TGTLOC[I,4];                              00018130  ARC07920
              K←NTGT: J←NDM+NDM+1;                                              00018140  ARC07920
              FOR1 J DO SAVEINE[K←K+1]←SDLOCDE[I];                              00018150  ARC07930
              K←NTGT+J;                                                         00018160  ARC07930
              FOR1 NTGLIST DO SAVEINE[K←K+1]←TOLISTE[I];                        00018170  ARC07940
              IF PACIN THEN                                                     00018180  ARC07950
               BEGIN                                                            00018190  ARC07950
                 COMMENT SET UP DMMOR1 AND DMMOR2;                              00018200  ARC07950
                 N3←3; FOR1 NDM DO INTA[I]←0;                                   00018210  ARC07960
                 TWOCOM←THREECOM←0;                                             00018220  ARC07960
                 FOR1 NPAC DO                                                   00018230  ARC07970
                  BEGIN                                                         00018240  ARC07970
                    KE←PACOTAN3+1; N3←N3+4; KA←PACOTAN3;                        00018250  ARC07970
                    FOR J←KE STEP 1 UNTIL KA DO INTE[J-KE]←PACDMS[J]            00018260  ARC07980
                    : KA←KA-KE;                                                 00018270  ARC07980
                    FOR1 KA DO                                                  00018280  ARC07990
                     BEGIN                                                      00018290  ARC07990
                       K←INTE[I];                                               00018300  ARC07990
```

```
      IF OCCUR(INTC,J,K,6,6,2)=J THEN                    00018310  ARC07990
       BEGIN                                             00018320  ARC07990
         PICKCHARS(K,KB,6,7,2);                          00018330  ARC08000
         INTA[KB]←INTA[KB]+1;                            00018340  ARC08000
       END;                                              00018350  ARC08000
    END;                                                 00018360  ARC08000
 END;                                                    00018370  ARC08000
KA←-1; KB←999;                                           00018380  ARC08020
FORI1 NDM DO                                             00018390  ARC08020
 BEGIN                                                   00018400  ARC08020
   KC←INTA[I]; IF KC<KB THEN KB←KC;                      00018410  ARC08020
   IF KA<KC THEN KA←KC;                                  00018420  ARC08030
 END;                                                    00018430  ARC08030
IF KB=0 THEN KB←1;                                       00018440  ARC08030
IF KA=KB THEN KE←KF←KA+1 ELSE IF KA-KB=1 THEN            00018450  ARC08050
 BEGIN                                                   00018460  ARC08050
   KE←KB; KF←KA+1                                        00018470  ARC08050
 END ELSE                                                00018480  ARC08050
 BEGIN                                                   00018490  ARC08050
   KE←KA; KF←99; K←1;                                    00018500  ARC08050
   FOR I←KB STEP 1 UNTIL KA DO                           00018510  ARC08060
    BEGIN                                                00018520  ARC08060
      KC←0;                                              00018530  ARC08060
      FORJ1 NDM DO IF INTA[J]≤I THEN KC←KC+1;            00018540  ARC08070
      IF KC>(K/3)×NDM THEN IF K=1 THEN                   00018550  ARC08090
       BEGIN                                             00018560  ARC08090
         K←2; KE←I                                       00018570  ARC08090
       END ELSE                                          00018580  ARC08090
       BEGIN                                             00018590  ARC08090
         KF←I; I←KA                                      00018600  ARC08090
       END;                                              00018610  ARC08090
    END FORI;                                            00018620  ARC08100
 END IFKA;                                               00018630  ARC08110
FORI1 NDM DO                                             00018640  ARC08120
 BEGIN                                                   00018650  ARC08120
   KA←INTA[I];                                           00018660  ARC08120
   IF KA>KE THEN SETBIT(THREECOR,I) ELSE IF KA>KE        00018670  ARC08130
   THEN SETBIT(TWOCOR,I);                                00018680  ARC08140
 END FORI1;                                              00018690  ARC08140
PICKCHARS(TWOCOR,PACDTA[0,0],1,1,8);                     00018700  ARC08150
PICKCHARS(THREECOR,PACDTA[0,1],1,1,8);                   00018710  ARC08160
PACDTA[0,2]←NPAC;                                        00018720  ARC08160
IF REBUILDPAC THEN                                       00018730  ARC08170
 BEGIN                                                   00018740  ARC08170
    COMMENT SETUP PACDTAF AND PACDMS ON DISK;            00018750  ARC08170
   N3←0;                                                 00018760  ARC08180
   FORI1 13 DO                                           00018770  ARC08180
    BEGIN                                                00018780  ARC08180
      FORJ 29 DO                                         00018790  ARC08180
       BEGIN                                             00018800  ARC08180
         MOVEWORD(PACDTAN3,INTC[J]);                     00018810  ARC08180
         N3←N3+1                                         00018820  ARC08190
       END;                                              00018830  ARC08190
      WRITE(PACDTAF ,30,INTC[*]);                        00018840  ARC08190
    END;                                                 00018850  ARC08190
   KA←29;                                                00018860  ARC08200
   FORI1 8 DO                                            00018870  ARC08200
    BEGIN                                                00018880  ARC08200
      WRITE(PACDMSF ,**,FOR J←KA-29 STEP 1 UNTIL KA      00018890  ARC08200
      DO PACDMS[J]);                                     00018900  ARC08210
      KA←KA+30                                           00018910  ARC08210
```

```
                    END                                                                    00018920  ARC08210
                  END                                                                      00018930  ARC08210
                END ELSE                                                                   00018940  ARC08210
                BEGIN                                                                      00018950  ARC08210
COMMENT SET UP PACDTA AND PACDMS FROM DISK;                                                00018960  ARC08220
                  N3←0;                                                                    00018970  ARC08230
                  FORI1 13 DO                                                              00018980  ARC08230
                   BEGIN                                                                   00018990  ARC08230
                     READ(PACDTAF ,30,INTC[*]);                                            00019000  ARC08230
                     FORJ 29 DO                                                            00019010  ARC08240
                      BEGIN                                                                00019020  ARC08240
                        MOVEWORD(INTC[J],PACDTAN3);                                        00019030  ARC08240
                        N3←N3+1                                                            00019040  ARC08240
                      END                                                                  00019050  ARC08240
                   END;                                                                    00019060  ARC08240
                  KA←29;                                                                   00019070  ARC08250
                  FORI1 8 DO                                                               00019080  ARC08250
                   BEGIN                                                                   00019090  ARC08250
                     READ(PACDMSF ,**,FOR J←KA-29 STEP 1 UNTIL KA DO                       00019100  ARC08260
                     PACDMS[J]); KA←KA+30                                                  00019110  ARC08260
                   END;                                                                    00019120  ARC08260
                  PICKCHARS(PACDTA[0,0],TWOCOR,1,1,8);                                     00019130  ARC08270
                  PICKCHARS(PACDTA[0,1],THREECOR,1,1,8);                                   00019140  ARC08280
                  NPAC←PACDTA[0,2];                                                        00019150  ARC08280
                                                                                           00019160  ARC08290
                END IF PACIN;                                                              00019170
               WRITE(DATAOUT[DBL],<"DMMOB1">);                                             00019180
               FORI1 NDM DO IF CHEKBIT(TWOCOR,I) THEN WRITE(DATAOUT,<                      00019190
               I3>,I); WRITE(DATAOUT[DBL]);                                                00019200
               WRITE(DATAOUT[DBL],<"DMMOB2">);                                             00019210
               FORI1 NDM DO IF CHEKBIT(THREECOR,I) THEN WRITE(                             00019220
               DATAOUT,<I3>,I);                                                            00019230  ARC08300
               IF REPEAT=999 THEN FORI1 NAM DO ATTACK[I,1]←(I DIV 5)+                      00019240  ARC08310
               1; IF CYCLE>1 OR COMPARUN THEN SETATTACK;                                   00019250  ARC08320
               IF WATE[1]=0 THEN                                                           00019260  ARC08320
                BEGIN                                                                      00019270  ARC08320
                  READ(PACDTAF[13],NFEATURES+1,REPWATE[*]);                                00019280  ARC08330
                  READ(PACDTAF[14],NFEATURES+1,WATE[*])                                    00019290  ARC08330
                END ELSE IF ADJWATE THEN                                                   00019300  ARC08330
                BEGIN                                                                      00019310  ARC08340
                  WRITE(PACDTAF[13],NFEATURES+1,REPWATE[*]);                               00019320  ARC08350
                  WRITE(PACDTAF[14],NFEATURES +1,WATE[*]);                                 00019330  ARC08350
                  REWIND(PACDTAF);                                                         00019340  ARC08350
                END;                                                                       00019350  ARC08360
               IF WATE[1]=0 THEN                                                           00019360  ARC08360
                BEGIN                                                                      00019370  ARC08360
                  WRITE(DATAOUT,<"WEIGHTS ARE ZERO">);                                     00019380  ARC08370
                  GO TO FINISH                                                             00019390  ARC08370
                END;                                                                       00019400  ARC08370
               WRITE(DATAOUT[DBL]);                                                        00019410  ARC08380
               WRITE(DATAOUT,<"INITIAL FEATURE WEIGHTS  ",8 F10.5/11                       00019420  ARC08390
               F10.5//>,FORI1 NFEATURES DO WATE[I]);                                       00019430  ARC08400
               WRITE(DATAOUT,<"INITIAL WEIGHT REPEATS   ",8 I10/11                         00019440  ARC08410
               I10//>,FORI1 NFEATURES DO REPWATE[I]);                                      00019450  ARC08420
               WRITE(DATAOUT,<"INITIAL VALUE OF TARGETS IS",F12.5///>                      00019460  ARC08420
               ,TOTVALUE);                                                                 00019470  ARC08430
             END PSEUDO BLOCK;                                                             00019480  ARC08451
            TME←TIME(1) ; LOCK(PACDTAF); LOCK(PACDMSF);                                    00019490  ARC08460
            IF NAM=0 THEN GO TO FINISH;                                                    00019500  ARC08461
                                                                                           00019510  ARC08461
COMMENT DATA RELEVANT TO THE CURRENT PERIOD IS ENTERED AT THIS POINT.                      00019520  ARC08462
        IT INCLUDES NEW AM ENTERING THE SYSTEM AND ANY PARAMETRIC DATA
```

```
        FOR DEFENSE OR ENVIRONMENT CONTROL. FOLLOWING DATA READIN, TABLE  00019530  ARC08463
        AMDATA AND ALTERS ARE GENERATED, PLANNING GROUPS ESTABLISHED      00019540  ARC08464
        AND HOUSEKEEPING DONE FOR TRAINING IF IN THE TRAINING MODE.       00019550  ARC08465
        FINALLY, IF HEURISTIC H7 IS ACTIVE THEN THE   "WHITES-OF-EYES"    00019560  ARC08466
        ALLOCATION  IS SELECTED AND BRANCE IS MADE TO SIMULATE RESULTS;   00019570  ARC08467
NEXTPERIOD:     PERIOD←PERIOD+1;                                          00019580  ARC08470
                BEGIN                                                     00019590  ARC08470
      COMMENT THIS PSEUDO BLOCK ENTERS AMS FOR CURRENT PERIOD;            00019600  ARC08480
LABEL             INTERREP,INTOVER,NEXT;                                  00019610  ARC08490
                  UNPACK(TYPATK,INTC);                                    00019620  ARC08500
                  WRITE(DATAOUT,<X1/"ACTIVE TARGETS ",8(X2,A1)/>,FORI     00019630  ARC08510
                  NTYPT-1 DO INTC[I]);                                    00019640  ARC08510
                  WRITE(DATAOUT,<"TIME TO PROCESS PERIOD",I4," IS",F8.2,  00019650  ARC08520
                  " SECONDS">, PERIOD-1,(TIME(1)-TME)/60);                00019660  ARC08530
                  TME←TIME(1); WRITE(DATAOUT[PAGE]);                      00019670  ARC08530
                  WRITE(DATAOUT[OBJ],<"ENTERING PERIOD",I4>,PERIOD);      00019680  ARC08540
                  IF BOOK2 THEN                                           00019690  ARC08550
                   BEGIN                                                  00019700  ARC08550
                     ANSWER←TRUE;                                         00019710  ARC08560
                     FORI 10 DO LBESTCOMB[I]←BOOKANSWER[PERIOD,I];        00019720  ARC08560
                     LEARNING←ANSWAS←ANSCOMB←FALSE;                       00019730  ARC08570
                     GO TO INTOVER;                                       00019740  ARC08580
                   END ELSE                                               00019750  ARC08590
                   BEGIN                                                  00019760  ARC08590
                     ANSWER←ANSWAS←ANSCOMB←FALSE;                         00019770  ARC08590
                     LEARNING←LSAVE                                       00019780  ARC08590
                   END;                                                   00019790  ARC08590
INTERREP:         READ(DATAIN [NO],A,CODE)[INTOVER];                      00019800  ARC08600
                  IF CODE="ANS" THEN                                      00019810  ARC08610
                   BEGIN                                                  00019820  ARC08610
                     READ(DATAIN,<X7,11(A3,X3)>,FORI 10 DO LBESTCOMB[I]); 00019830  ARC08620
                     ; ANSWER←TRUE; LEARNING←FALSE;                       00019840  ARC08620
                     FORI 10 DO IF LBESTCOMB[I]≥64 THEN SETBIT(           00019850  ARC08630
                     LBESTCOMB[I],28);                                    00019860  ARC08630
                     IF BOOK1 THEN FORI 10 DO BOOKANSWER[PERIOD,I]←       00019870  ARC08640
                     LBESTCOMB[I];                                        00019880  ARC08640
                     WRITE(DATAOUT,<"ANSWER SPECIFIED">);                 00019890  ARC08650
                     GO TO INTERREP                                       00019900  ARC08650
                   END;                                                   00019910  ARC08650
                  IF CODE="PER" THEN                                      00019920  ARC08660
                   BEGIN                                                  00019930  ARC08660
                     READ(DATAIN[NO],/,I,I);                              00019940  ARC08660
                     IF PERIOD=I THEN                                     00019950  ARC08660
                      BEGIN                                               00019960  ARC08660
                        READ(DATAIN);                                     00019970  ARC08670
                        GO TO INTERREP                                    00019980  ARC08670
                      END ELSE GO TO INTOVER;                             00019990  ARC08670
                   END;                                                   00020000  ARC08670
                  IF CODE="END" THEN                                      00020010  ARC08680
                   BEGIN                                                  00020020  ARC08680
                     READ(DATAIN); GO TO INTOVER                          00020030  ARC08680
                   END;                                                   00020040  ARC08680
                  IF CODE="TDM" THEN                                      00020050  ARC08690
                   BEGIN                                                  00020060  ARC08690
                     READ(DATAIN,/,I,I,DLIST[I]);                         00020070  ARC08690
                     GO TO INTERREP                                       00020080  ARC08690
                   END;                                                   00020090  ARC08690
                  IF CODE="DM " THEN                                      00020100  ARC08700
                   BEGIN                                                  00020110  ARC08700
                     READ(DATAIN,/,I,I,SDLOC0[I×2],SDLOC0[I×2+1]);        00020120  ARC08700
                     GO TO INTERREP                                       00020130  ARC08710
```

```
 END;                                                                    00020140        ARC08710
IF CODE="GRO" THEN                                                       00020150        ARC08720
 BEGIN                                                                   00020160        ARC08720
   READ(DATAIN,/,I,GRPCUTOFF);                                           00020170        ARC08720
   GO TO INTERREP                                                        00020180        ARC08720
 END;                                                                    00020190        ARC08720
IF CODE="ACT" THEN                                                       00020200        ARC08730
 BEGIN                                                                   00020210        ARC08730
   READ(DATAIN,/,I,ACTIVEINDEX);                                         00020220        ARC08730
   GO TO INTERREP                                                        00020230        ARC08730
 END;                                                                    00020240        ARC08730
IF CODE="TYP" THEN                                                       00020250        ARC08740
 BEGIN                                                                   00020260        ARC08740
   READ(DATAIN,<X13,B(A1,X2)>,FOR I 7 DO INTC[I]);                       00020270        ARC08740
   PACKWORD(TYPATA,INTC);                                                00020280        ARC08750
   GO TO INTERREP                                                        00020290        ARC08750
 END;                                                                    00020300        ARC08750
IF CODE="HEU" THEN                                                       00020310        ARC08760
 BEGIN                                                                   00020320        ARC08760
   SETHEUR; GO TO INTERREP                                               00020330        ARC08760
 END;                                                                    00020340        ARC08760
IF CODE="MIN" THEN                                                       00020350        ARC08770
 BEGIN                                                                   00020360        ARC08770
   READ(DATAIN,/,I,MINTGTVAL);                                           00020370        ARC08770
   GO TO INTERREP                                                        00020380        ARC08770
 END;                                                                    00020390        ARC08770
IF CODE="OUT" THEN                                                       00020400        ARC08780
 BEGIN                                                                   00020410        ARC08780
   READ(DATAIN,/,K,K);                                                   00020420        ARC08780
   IF K=1 THEN OUT1←TRUE ELSE IF K=2 THEN OUT2←TRUE                      00020430        ARC08790
   ELSE IF K=3 THEN OUT3←TRUE ELSE IF K=4 THEN OUT1←                     00020440        ARC08800
   FALSE ELSE IF K=5 THEN OUT2←FALSE ELSE IF K=6 THEN                    00020450        ARC08810
   OUT3←FALSE ELSE IF K=7 THEN OUT7←TRUE ELSE OUT7←                      00020460        ARC08810
   FALSE; GO TO INTERREP                                                 00020470        ARC08811
 END;                                                                    00020480        ARC08811
IF CODE="LEA" THEN                                                       00020490        ARC08820
 BEGIN                                                                   00020500        ARC08820
   READ(DATAIN);                                                         00020510        ARC08820
   WRITE(DATAOUT,<"COMMENCE LEARNING" >);                                00020520        ARC08830
   LSAVE←LEARNING←TRUE; GO TO INTERREP                                   00020530        ARC08830
 END;                                                                    00020540        ARC08830
IF CODE="STO" THEN                                                       00020550        ARC08840
 BEGIN                                                                   00020560        ARC08840
   READ(DATAIN);                                                         00020570        ARC08840
   WRITE(DATAOUT,<"STOP LEARNING">);                                     00020580        ARC08840
   LSAVE←LEARNING←FALSE;                                                 00020590        ARC08850
   GO TO INTERREP                                                        00020600        ARC08850
 END;                                                                    00020610        ARC08850
IF CODE="TAR" THEN                                                       00020620        ARC08860
 BEGIN                                                                   00020630        ARC08860
   READ(DATAIN,<X7,A6,X6,R6,0>,KA,T);                                    00020640        ARC08860
   FOR I1 NOW DO IF KA=TGTTAB[I,0] THEN                                  00020650        ARC08870
    BEGIN                                                                00020660        ARC08870
      TA←TGTICV[I,4];                                                    00020670        ARC08870
      IF T<TA THEN                                                       00020680        ARC08870
       BEGIN                                                             00020690        ARC08870
         TGTICV[I,4]←T;                                                  00020700        ARC08880
         TOTVALUE←TOTVALUE+T-TA;                                         00020710        ARC08880
       END;                                                              00020720        ARC08880
      GO TO INTERREP                                                     00020730        ARC08880
    END                                                                  00020740        ARC08880
```

```
                    END;                                                    00020750        ARC08880
                   IF CODE="LAS" OR CODE="RUN" OR CODE="NAT" THEN GO TO     00020760        ARC08890
                   INTOVER;                                                 00020770        ARC08890
                   WRITE(DATAOUT,<"BAD CARD CODE ",A3," CARD DISREGARDED"   00020780        ARC08900
                   >,CODE); READ(DATAIN); GO TO INTERRP;                    00020790        ARC08910
INTOVER:           FORI NTYPE=1 DO PICKCHARS(SDMV[I],DMVOID[I],1,1,8);      00020800        ARC08930
                   FORI NTDLIST DO TDLIST[I]←DLIST[I]; K←1;                 00020810        ARC08940
                   FORI1 NDM DO FOR J←3 STEP 1 UNTIL NTYPE+2 DO DMLOCO[I,   00020820        ARC08950
                   J]←SDLOCO[K←K+1];                                        00020830        ARC08950
                   FOR I←CURATT STEP 1 UNTIL NAM DO IF ATTACK[I,1]≤         00020840        ARC08960
                   PERIOD THEN                                              00020850        ARC08960
                    BEGIN                                                   00020860        ARC08960
                      AMNOW←AMNOW+1; AMDET←AMDET+1;                         00020870        ARC08970
                      NEWAM(PERIOD,I); CURATT←I+1;                          00020880        ARC08980
                    END ELSE I←NAM; FINALT←-1; FIRSTPASS←TRUE;              00020890        ARC08990
                   CW←CWO←1;                                                00020900        ARC09000
                   FORI1 MAXAMDT DO IF AMDATA[I,0]>0 THEN IF HEURISTICS     00020910        ARC09020
                   AND TGTLOV[AMDATA[I,3],4]≤MINTGTVAL THEN                 00020920        ARC09020
                    BEGIN                                                   00020930        ARC09020
                      WRITE(DATAOUT,<"TARGET ",A6,                          00020940        ARC09030
                      " VALUE BELOW MINIMUM, AM",I4," IGNORED"/>, AMDATA[   00020950        ARC09040
                      I,4],AMDATA[I,1]); AMDATA[I,0]←0;                     00020960        ARC09050
                      K←AMDATA[I,7];                                        00020970        ARC09050
                      IF K≠0 THEN AMDATA[I,K]←0;                            00020980        ARC09050
                    END ELSE                                                00020990        ARC09050
                    BEGIN                                                   00021000        ARC09050
                      GENALTERS(I); FIRSTPASS←FALSE;                        00021010        ARC09060
                    END;                                                    00021020        ARC09060
                   LASTALT←FINALT; WRITE(DATAOUT[DBL]);                     00021030        ARC09090
                   WRITE(DATAOUT, COMBIN,CW,CWO);                           00021040        ARC09090
                                                                            00021050        ARC09100
COMMENT SORT ALTERS,AM ENGAGEABLE IN CURR PERIOD IN LOW ORDER POSITION;     00021060        ARC09100
                   KA←0;                                                    00021070        ARC09110
                   FORI LASTALT DO IF ALTERS[I,1]≠0 THEN                    00021080        ARC09110
                    BEGIN                                                   00021090        ARC09110
                      FORJ 11 DO                                            00021100        ARC09110
                       BEGIN                                                00021110        ARC09110
                         KB←ALTERS[KA,J];                                   00021120        ARC09120
                         ALTERS[KA,J]←ALTERS[I,J];                          00021130        ARC09120
                         ALTERS[I,J]←KB;                                    00021140        ARC09120
                       END;                                                 00021150        ARC09120
                      KA←KA+1                                               00021160        ARC09120
                    END;                                                    00021170        ARC09120
                   FUTALT←KA; BPAC←FALSE;                                   00021180        ARC09130
                   IF FIRSTPASS THEN IF CURATT≤NAM THEN GO TO NEXTPERIOD    00021190        ARC09140
                   ELSE GO TO FINISH;                                       00021200        ARC09140
                   IF FUTALT=0 THEN GO TO STMRESULT;                        00021210        ARC09150
                   IF LEARNING THEN                                         00021220        ARC09160
                    BEGIN                                                   00021230        ARC09160
                      NEXTALT←FINALT+1; TREPEAT←1;                          00021240        ARC09160
                      WRITE(DATAOUT,<//                                     00021250        ARC09170
                      "NEXTPERIOD AMS FOR LEARNING PROCESS"//>);            00021260        ARC09170
                      FOR I←CURATT STEP 1 UNTIL NAM DO IF ATTACK[I,1]≤      00021270        ARC09180
                      PERIOD+1 THEN                                         00021280        ARC09180
                       BEGIN                                                00021290        ARC09180
                         AMNOW←AMNOW+1; AMDET←AMDET+1;                      00021300        ARC09190
                         NEWAM(PERIOD+1,I);                                 00021310        ARC09190
                         IF NOTGT OR NODEFENSE OR TOOLATE THEN K←K ELSE     00021320        ARC09200
                         GENALTERS(LASTPROW);                               00021330        ARC09200
                         CURATT←I+1                                         00021340        ARC09210
                       END ELSE I←NAM;                                      00021350        ARC09210
```

```
   WRITE(DATAOUT,COMBIN,CW,CWO);                                    00021360        ARC09220
   WRITE(DATAOUT[DBL]);                                             00021370        ARC09220
 END IF LEARNING;                                                   00021380        ARC09230
IF CW<GRPCUTOFF THEN FORI LASTALT DO ALTERS[I,11]←0                 00021390        ARC09231
ELSE IF HEURISTIC6 THEN FORI LASTALT DO ALTERS[I,11]←               00021400        ARC09240
GROUPS[ALTERS[I,5],[30:12]] ELSE IF HEURISTIC5 THEN                 00021410        ARC09260
FINDGROUPS(0,LASTALT) ELSE                                          00021420        ARC09260
 BEGIN                                                              00021430        ARC09260
   FORI LASTALT DO ALTERS[I,11]←0;                                  00021440        ARC09270
   NUMGRPS←0;                                                       00021450        ARC09270
 END;                                                               00021460        ARC09270
IF OUT1 THEN                                                        00021470        ARC09280
 BEGIN                                                              00021480        ARC09280
   FORI FINALT DO                                                   00021490        ARC09290
    BEGIN                                                           00021500        ARC09290
      WRITE(DATAOUT[DBL]);                                          00021510        ARC09290
      WRITE(DATAOUT,<3 I10>,AMDATA[ALTERS[I,0],1],                  00021520        ARC09300
      ALTERS[I,0],ALTERS[I,2]);                                     00021530        ARC09300
      KE←ALTERS[I,3];                                               00021540        ARC09310
      FOR J← 5 STEP 1 UNTIL KE DO                                   00021550        ARC09310
       BEGIN                                                        00021560        ARC09310
         BREAKDMWORD(ALTERS[I,J],KA,KB,KC,KD);                      00021570        ARC09320
         WRITE(DATAOUT,<4 I10>,KA,KB,KC,KD);                        00021580        ARC09330
       END;                                                         00021590        ARC09330
      WRITE(DATAOUT,<I10>,ALTERS[I,11])                             00021600        ARC09340
    END;                                                            00021610        ARC09340
   WRITE(DATAOUT[PAGE]);                                            00021620        ARC09340
 END OUT1;                                                          00021630        ARC09340
IF HEURISTIC7 THEN                                                  00021640        ARC09360
 BEGIN                                                              00021650        ARC09360
       COMMENT APPLY HEURISTIC 7 TO SELECT DMS;                     00021660        ARC09360
   FORI FUTALT=1 DO                                                 00021670        ARC09370
    BEGIN                                                           00021680        ARC09370
      BESTALT[I]←0; KA←ALTERS[I,1];                                 00021690        ARC09370
      IF ALTERS[I,KA]=0 THEN                                        00021700        ARC09380
       BEGIN                                                        00021710        ARC09380
         ALTX←I; ALTERS[I,4]←KA; SELECTDM;                          00021720        ARC09380
         IF FEAS THEN                                               00021730        ARC09390
          BEGIN                                                     00021740        ARC09390
            BESTALT[I]←ALTERS[I,KA];                                00021750        ARC09390
            GO TO NEXT                                              00021760        ARC09390
          END ELSE IF KA=5 THEN GO TO NEXT ELSE KA←KA-              00021770        ARC09400
         1                                                          00021780        ARC09400
       END DM PART;                                                 00021790        ARC09400
      KB←K←0;                                                       00021800        ARC09400
      FOR J←5 STEP 1 UNTIL KA DO                                    00021810        ARC09410
       BEGIN                                                        00021820        ARC09410
         BREAKDMWORD(ALTERS[I,J],KE,KE,KF,KG);                      00021830        ARC09410
         KD←DMLOC[KF,KG+3];                                         00021840        ARC09420
         IF KD>KB THEN                                              00021850        ARC09420
          BEGIN                                                     00021860        ARC09420
            KB←KD; KC←J;                                            00021870        ARC09420
            K ←0                                                    00021880        ARC09420
          END ELSE IF KD≠0 AND KD=KB THEN                           00021890        ARC09430
          BEGIN                                                     00021900        ARC09430
            INTC[K]←J;                                              00021910        ARC09430
            K←K+1                                                   00021920        ARC09430
          END                                                       00021930        ARC09430
       END FOR J LOOP;                                              00021940        ARC09430
      IF KB=0 THEN GO TO NEXT; INTC[K]←KC;                          00021950        ARC09440
      KE←INTC[ENTIER(RANDOM×(K+1))];                                00021960        ARC09440
```

```
                        BESTALT[I]←ALTERS[I,KE];                                00021970        ARC09450
                        BREAKDMWORD(BESTALT[I],KF,KF,KF,KG);                    00021980        ARC09460
                        DMLOCO[KF,KG+3]←DMLOCO[KF,KG+3]-1;                      00021990        ARC09460
NEXT:                  END FOR I;                                               00022000        ARC09470
                      GO TO SIMRESULT                                           00022010        ARC09470
                    END HEURISTIC PROCESS;                                      00022020        ARC09470
                  END PSEUDO BLOCK;                                             00022030        ARC09480
                                                                                00022040        ARC09500
COMMENT DECISION ROUTINE USING THE DECISION FUNCTION.                           00022050        ARC09500
        THIS PORTION OF THE PROGRAM DETERMINES THE ALLOCATION MINIMIZING        00022060        ARC09501
        THE DECISION FUNCTION UNDER CONSTRAINTS IMPOSED BY ACTIVE HEURIS        00022070        ARC09502
        TICS. IF IN THE TRAINING MODE THE LEARNING PROCESS IS APPLIED;          00022080        ARC09503
              BEGIN                                                             00022090        ARC09503
               COMMENT DECISION ROUTINE USING DECISION FUNCTION;                00022100        ARC09500
LABEL           D1,D2,D21,D3,D4,D5,DFIN,DFEAS,L11,L12;                          00022110        ARC09510
BOOLEAN         FLIP,REPFLAG;                                                   00022120        ARC09510
DEFINE          RESETGRP= KA←ALTERS[I,4];                                       00022130        ARC09511
                IF KA≠J THEN                                                    00022140        ARC09511
                 BEGIN                                                          00022150        ARC09511
                   BREAKDMWORD(ALTERS[ I,KA],KB,KB,KC,KD);                      00022160        ARC09512
                   IF KB≠0 THEN                                                 00022170        ARC09512
                    BEGIN                                                       00022180        ARC09512
                      DMLOCO[KC,KD+3]←KB←DMLOCO[KC,KD +3]+1;                    00022190        ARC09513
                      IF KB=1 THEN CLEARBIT(DMVOID[KD],KC);                     00022200        ARC09513
                    END ELSE                                                    00022210        ARC09513
                    BEGIN                                                       00022220        ARC09513
                      TDLIST[KD]← KB←TDLIST[KD]+1;                              00022230        ARC09514
                      ALTERS[I,KA]←0                                            00022240        ARC09514
                    END;                                                        00022250        ARC09514
                   BREAKDMWORD(BESTCOMB[I],KB,KB,KC, KD);                       00022260        ARC09515
                   IF KB=0 THEN TDLIST[KD]←TDLIST[KD]-1 ELSE                    00022270        ARC09515
                    BEGIN                                                       00022280        ARC09515
                      DMLOCO[KC,KD+3]← KB←DMLOCO[KC,KD+3]-1;                    00022290        ARC09516
                      IF KB=0 THEN SETBIT( DMVOID[KD],KC)                       00022300        ARC09516
                    END;                                                        00022310        ARC09516
                 END;                                                           00022320        ARC09516
                ALTERS[1,4]←J; J←10*;                                           00022330        ARC09517
                FOR I FUTALT-1 DO BESTALT[I]←0;                                 00022340        ARC09520
                FOR NFEATURES DO TFEAT[I]←XFEAT[I]←0;                           00022350        ARC09520
                FLAG←REPFLAG←BPAC←FALSE; TREPEAT←1;                             00022360        ARC09530
                LV1←LV2←LV3←0;                                                  00022370        ARC09530
                IF ANSWER THEN FOR I LASTALT DO                                 00022380        ARC09540
                 BEGIN                                                          00022390        ARC09540
                   ANSINDEX[I]←K← OCCUR(ALTERS[I,5],6,LBESTCOMB[I],6,           00022400        ARC09550
                   6,2) +5;                                                     00022410        ARC09550
                   IF K=11 THEN                                                 00022420        ARC09560
                    BEGIN                                                       00022430        ARC09560
                      ANSWER←FALSE;                                             00022440        ARC09560
                      WRITE(DATAOUT,<"ANSWER IS INFEASIBLE">)                   00022450        ARC09570
                    END                                                         00022460        ARC09570
                 END;                                                           00022470        ARC09570
                INITIALALT(0,LASTALT,TRUE);                                     00022480        ARC09580
                IF NOT FEAS THEN INITIALALT(0,LASTALT,FALSE );                  00022490        ARC09590
                IF HEURISTIC1 THEN PACEVAL ELSE FOR I NFEATURES DO              00022500        ARC09590
                PFEAT[I]←TFEAT[I];                                              00022510        ARC09590
                FOR I NFEATURES DO                                              00022520        ARC09600
                 BEGIN                                                          00022530        ARC09600
                   INFEAT[I]←FEATURE[I]←TFEAT[I];                               00022540        ARC09600
                   LPFEATURE[I]← PFEATURE[I]←INPFEAT[I]←PFEAT[I]                00022550        ARC09610
                 END;                                                           00022560        ARC09610
                FOR I FUTALT-1 DO BESTALT[I]←IF ALTERS[I,4]<ALTERS[I,           00022570        ARC09620
```

```
                1] THEN ALTERS[I, ALTERS[I,4]] ELSE 0;                    00022560        ARC09630
                IF LEARNING THEN                                          00022590        ARC09640
                 BEGIN                                                    00022600        ARC09640
                   LOOKAHEAD;                                             00022610        ARC09640
                   FORI NFEATURES DO INLFEAT[I]←LFEATURE[I]               00022620        ARC09650
                 END;                                                     00022630        ARC09650
D1:             ALTX←LASTALT;                                             00022640        ARC09660
                IF REPFLAG THEN                                           00022650        ARC09660
                 BEGIN                                                    00022660        ARC09660
                   REPFLAG←FALSE;                                         00022670        ARC09660
                   FORI NTYPE-1 DO PICKCHARS(SDMV[I],DMVOID[I],1,1,8);    00022680        ARC09670
                   FORI NTDLIST DO TDLIST[I]←DLIST[I]; K←1;               00022690        ARC09680
                   FORI1 NDM DO FOR J←3 STEP 1 UNTIL NTYPE+2 DO           00022700        ARC09690
                   DMLOCO[I,J]←SDLOCO[K←K+1];                             00022710        ARC09690
                   FORI LASTALT DO                                        00022720        ARC09700
                    BEGIN                                                 00022730        ARC09700
                      KA←ALTERS[I,3]; KB←ALTERS[I,1];                     00022740        ARC09700
                      FOR J←5 STEP 1 UNTIL KA DO                          00022750        ARC09710
                       BEGIN                                              00022760        ARC09710
                         BREAKDMWORD(ALTERS[I,J],K,K,KD,KC);              00022770        ARC09710
                         IF K=0 THEN IF TDLIST[KC]=0 THEN GO TO L11       00022780        ARC09720
                         ELSE TDLIST[KC]←TDLIST[KC]-1 ELSE IF CHEKBIT(    00022790        ARC09730
                         DMVOID[KC ],KD) THEN GO TO L11 ELSE              00022800        ARC09730
                          BEGIN                                           00022810        ARC09730
                            K←DMLOCO[KD,KC+3]←DMLOCO[KD,KC+3]-1;          00022820        ARC09740
                            IF K=0 THEN SETBIT(DMVOID[KC],KD)             00022830        ARC09740
                          END;                                            00022840        ARC09740
                         ALTERS[I,4]←J;                                   00022850        ARC09750
                         BESTALT[I]←IF J<KA THEN ALTERS[I,J] ELSE 0;      00022860        ARC09750
                         GO TO L12;                                       00022870        ARC09760
L11:                    END FOR J;                                        00022880        ARC09760
L12:                    IF ALTERS[I,4]≠KA THEN ALTERS[I,KA]←0;            00022890        ARC09770
                      END FORI;                                           00022900        ARC09780
                   FORI NFEATURES DO                                      00022910        ARC09790
                    BEGIN                                                 00022920        ARC09790
                      PFEATURE[I] ←INPFEAT[I];                            00022930        ARC09790
                      FEATURE[I]← TFEAT[I]←INFEAT[I];                     00022940        ARC09800
                      LFEATURE[I]←INLFEAT[I];                             00022950        ARC09800
                    END;                                                  00022960        ARC09800
                   MULT(WATE,INLFEAT,LV2,I,NFEATURES);                    00022970        ARC09810
                   LV1←INLFEAT[4]; LV3←INLFEAT[3];                        00022980        ARC09810
                 END IF REPFLAG;                                          00022990        ARC09820
                MULT(WATE,INPFEAT,V2,I,NFEATURES);                        00023000        ARC09830
                V1←INPFEAT[4]; V3←INPFEAT[3];                             00023010        ARC09830
                PFEATURE[0]←V2;                                           00023020        ARC09840
                IF NOT ANSWAS THEN FORI NFEATURES DO LPFEATURE[I]←        00023030        ARC09850
                PFEATURE[I];                                              00023040        ARC09850
                IF ANSWER THEN                                            00023050        ARC09860
                 BEGIN                                                    00023060        ARC09860
                   FORI LASTALT DO IF ANSINDEX[I]≠ALTERS[I,4] THEN I←     00023070        ARC09870
                   999; IF I<999 THEN ANSWAS←TRUE                         00023080        ARC09870
                 END;                                                     00023090        ARC09870
                IF HEURISTICO THEN LV1←LV3←V1←V3←0;                       00023100        ARC09880
                FORI LASTALT DO BESTCOMB[I]←ALTERS[I,ALTERS[I,4]];        00023110        ARC09890
                IF NOT ANSWER THEN FORI LASTALT DO LBESTCOMB[I]←          00023120        ARC09900
                BESTCOMB[I];                                              00023130        ARC09900
                IF OUT2 THEN                                              00023140        ARC09910
                 BEGIN                                                    00023150        ARC09910
                   WRITE(DATAOUT,<"BESTCOMB",30(X1,A2)/"BESTALT",30(      00023160        ARC09920
                   X1,A2)/>,FORI LASTALT DO BESTCOMB[I],FORI FUTALT-1      00023170        ARC09930
                   DO BESTALT[I]);                                        00023180        ARC09930
```

```
                    WRITE(DATAOUT,<10 F10.4 />,FORI NFEATURES DO          00023190        ARC09940
                    INPFEAT[I]);                                          00023200        ARC09940
                  END;                                                    00023210        ARC09940
                 GRPNOW←0;                                                00023220        ARC09950
D2:              IF ALTX<0 THEN IF GRPNOW=NUMGRPS THEN GO TO DFIN ELSE    00023230        ARC09960
                  BEGIN                                                   00023240        ARC09960
                    FORI LASTALT DO IF ALTERS[I,11]=GRPNOW THEN FOR J←    00023250        ARC09961
                    5 STEP 1 UNTIL 10 DO IF ALTERS[I,J]=RESTCOMB[I]       00023260        ARC09962
                    THEN                                                  00023270        ARC09962
                     BEGIN                                                00023280        ARC09962
                       RESETGRP                                           00023290        ARC09962
                     END ELSE IF ALTERS[I,J]=0 THEN                       00023300        ARC09963
                     BEGIN                                                00023310        ARC09963
                       ALTERS[I,J]←RESTCOMB[I];                           00023320        ARC09963
                       RESETGRP                                           00023330        ARC09963
                     END;                                                 00023340        ARC09963
                    FORI NFEATURES DO TFEAT[I]←FEATURE[I];                00023350        ARC09964
                    ALTX←LASTALT; GRPNOW←GRPNOW+1; GO TO D2;              00023360        ARC09970
                  END;                                                    00023370        ARC09970
                 IF ALTERS[ALTX,11]≠GRPNOW THEN                           00023380        ARC09980
                  BEGIN                                                   00023390        ARC09980
                    ALTX←ALTX-1; GO TO D2                                 00023400        ARC09980
                  END;                                                    00023410        ARC09980
                 KC←ALTERS[ALTX,2]; KA←ALTERS[ALTX,4];                    00023420        ARC09990
                 IF KC<KA THEN KC←KA; FLIP←FALSE;                         00023430        ARC09990
                 IF KC=5 THEN                                             00023440        ARC10000
                  BEGIN                                                   00023450        ARC10000
                    ALTX←ALTX-1; GO TO D2                                 00023460        ARC10000
                  END;                                                    00023470        ARC10000
                 IF KA<KC THEN KB←KA+1 ELSE                               00023480        ARC10010
                  BEGIN                                                   00023490        ARC10010
                    KB←5; FLIP←TRUE                                       00023500        ARC10010
                  END;                                                    00023510        ARC10010
                 KE←ALTERS[ALTX,KA];                                      00023520        ARC10020
D21:             KD←ALTERS[ALTX,KB];                                      00023530        ARC10020
                 IF KD=0 THEN                                             00023540        ARC10030
                  BEGIN                                                   00023550        ARC10030
                    ALTERS[ALTX,4]←KB; SELECTTDM;                         00023560        ARC10030
                    ALTERS[ALTX,4]←KA;                                    00023570        ARC10030
                    IF NOT FEAS THEN GO TO DFEAS                          00023580        ARC10040
                  END ELSE                                                00023590        ARC10040
                  BEGIN                                                   00023600        ARC10040
                    BREAKDMWORD(KD,KF,KF,KF, KG);                         00023610        ARC10050
                    IF CHEKBIT(DMVOID[KG],KF) THEN GO TO DFEAS            00023620        ARC10050
                  END;                                                    00023630        ARC10050
                 IF KE=0 THEN                                             00023640        ARC10060
                  BEGIN                                                   00023650        ARC10060
                    FLAG←TRUE; GO TO D3                                   00023660        ARC10060
                  END;                                                    00023670        ARC10060
                 IF KE<@4 THEN                                            00023680        ARC10070
                  BEGIN                                                   00023690        ARC10070
                    SELECTTDM; GO TO D3;                                  00023700        ARC10070
                  END;                                                    00023710        ARC10070
D3:              ALTERS[ALTX,4]←KB; EVALUATE(KA,KB);                      00023720        ARC10080
                 IF FLAG THEN                                             00023730        ARC10090
                  BEGIN                                                   00023740        ARC10090
                    T←TGTLCV[AMDATA[ALTERS[ALTX,0],3],4];                 00023750        ARC10090
                    TFEAT[9]←TFEAT[9]-T; FLAG←FALSE                       00023760        ARC10100
                  END;                                                    00023770        ARC10100
                 MULT(WATE,TFEAT,TV2,I,NFEATURES);                        00023780        ARC10110
                 TV1←TFEAT[4]; TV3←TFEAT[3];                              00023790        ARC10110
```

```
                IF HEURISTIC9 THEN TV1←TV3←0; TFEAT[0]←TV2;                    00023800  ARC10120
                IF ANSWER AND NOT ANSWAS THEN                                  00023810  ARC10130
                 BEGIN                                                         00023820  ARC10130
LABEL              L1;                                                         00023830  ARC10130
                   FORI LASTALT DO IF ANSINDEX[I]≠ALTERS[I,4] THEN GO          00023840  ARC10140
                   TO L1; ANSWAS←ANSCOMB←TRUE;                                 00023850  ARC10150
L1:              END;                                                          00023860  ARC10160
                IF NOT HEURISTIC9 THEN FORI NFEATURES DO PFEAT[I]←             00023870  ARC10170
                TFEAT[I] ELSE IF ANSWER OR LEARNING THEN PACEVAL ELSE          00023880  ARC10180
                IF V1<TV1 THEN K←K ELSE IF TV1<V1 OR TV2+@-5<V2 THEN           00023890  ARC10190
                PACEVAL;                                                       00023900  ARC10190
                IF ANSCOMB THEN                                                00023910  ARC10200
                 BEGIN                                                         00023920  ARC10200
                   FORI NFEATURES DO LPFEATURE[I]←PFEAT[I];                    00023930  ARC10200
                   ANSCOMB←FALSE;                                              00023940  ARC10210
                 END;                                                          00023950  ARC10210
                IF TV1<V1 THEN GO TO D4;                                       00023960  ARC10220
                IF V1<TV1 OR V2<TV2+@-5 THEN GO TO D5;                         00023970  ARC10220
                IF TV2<V2 OR TV3<V3 THEN GO TO D4 ELSE GO TO D5;               00023980  ARC10230
D4:             FORI NFEATURES DO PFEATURE[I]←PFEAT[I];                        00023990  ARC10240
                V1←TV1; V2←TV2; V3←TV3;                                        00024000  ARC10240
                FORI NFEATURES DO FEATURE[I]←TFEAT[I];                         00024010  ARC10250
                FORI LASTALT DO                                                00024020  ARC10260
                 BEGIN                                                         00024030  ARC10260
                   KB←ALTERS[I,4];                                             00024040  ARC10260
                   KA←BESTCOMB[I]←ALTERS[I,KB];                                00024050  ARC10270
                   BESTALT[I]←IF KB≤ALTERS[I,1] THEN KA ELSE 0                 00024060  ARC10280
                 END;                                                          00024070  ARC10280
                IF OUT2 THEN                                                   00024080  ARC10290
                 BEGIN                                                         00024090  ARC10290
                   WRITE(DATAOUT,<"BESTCOMB",30(X1,A2)/"BESTALT",30(           00024100  ARC10300
                   X1,A2)/>,FORI LASTALT DO BESTCOMB[I],FORI FUTALT-1          00024110  ARC10310
                   DO BESTALT[I]);                                             00024120  ARC10310
                   WRITE(DATAOUT,<10 F10.4 />,FORI NFEATURES DO PFEAT[         00024130  ARC10320
                   I]);                                                        00024140  ARC10320
                 END;                                                          00024150  ARC10320
D5:             IF LEARNING AND NEXTALT≤FINALT THEN LOOKAHEAD;                 00024160  ARC10330
                IF FLIP THEN ALTX←ALTX-1 ELSE ALTX←LASTALT;                    00024170  ARC10340
                GO TO D2;                                                      00024180  ARC10340
DFEAS:          KB←IF KB<KC THEN KB+1 ELSE 5;                                  00024190  ARC10350
                IF KB=KA THEN                                                  00024200  ARC10360
                 BEGIN                                                         00024210  ARC10360
                   ALTX←ALTX-1; GO TO D2;                                      00024220  ARC10360
                 END;                                                          00024230  ARC10360
                IF KB=5 THEN FLIP←TRUE; GO TO D21;                             00024240  ARC10370
DFIN:           IF ANSWER AND NOT ANSWAS THEN WRITE(DATAOUT,<                  00024250  ARC10380
                "ANSWER NOT FOUND">);                                          00024260  ARC10380
                                                                               00024270  ARC10380
COMMENT AT THIS POINT THE ALLOCATION HAS BEEN SELECTED. IF THIS IS NOT A       00024280  ARC10380
        TRAINING RUN THEN THE NEXT SECTION IS SKIPPED AND RESULTS SIMU-        00024290  ARC10381
        LATED OTHERWISE IF THE ALLOCATION AND KEY ALLOCATION DIFFER (IE        00024300  ARC10382
        PFEATURE AND LPFEATURE GIVE DIFFERENT DECISION FUNCTION VALUES)        00024310  ARC10383
        THE FEATURE WEIGHTS ARE ADJUSTED AND THE DECISION ROUTINE REPEAT       00024320  ARC10384
        ED UNTIL THEY AGREE OR REPEAT=TREPEAT;                                 00024330  ARC10385
                IF ANSWAS OR(LEARNING AND NEXTALT≤FINALT) THEN                 00024340  ARC10390
                 BEGIN                                                         00024350  ARC10390
                   IF HEURISTIC9 THEN LV1←LV3←0 ELSE                           00024360  ARC10400
                    BEGIN                                                      00024370  ARC10400
                      LV1←LPFEATURE[4];                                        00024380  ARC10400
                      LV3←LPFEATURE[ 3];                                       00024390  ARC10410
                    END;                                                       00024400  ARC10410
```

```
                    MULT(WATE,LPFEATURE,LV2,[,NFEATURES);                          00024410  ARC10410
                    LPFEATURE[0]←LV2; REPFLAG←FALSE;                               00024420  ARC10420
                    IF V2<LV2 THEN                                                 00024430  ARC10430
                     BEGIN                                                         00024440  ARC10430
                       NEWWATE←TRUE;                                               00024450  ARC10440
                       FOR I1 NFEATURES DO                                         00024460  ARC10440
                        BEGIN                                                      00024470  ARC10440
                          TA←PFEATURE[I]; TB←LPFEATURE[I];                         00024480  ARC10450
                          XWATE[I]←WATE[I]×(IF ABS(TA)<@-5 THEN IF ABS(            00024490  ARC10450
                          TB)<@-5 THEN 1 ELSE IF TB<0 THEN 2 ELSE .5               00024500  ARC10460
                          ELSE IF ABS(TB)<@-5 THEN IF TA<0 THEN .5                 00024510  ARC10470
                          ELSE 2 ELSE IF TA<0 THEN IF TB<0 THEN TB/TA              00024520  ARC10470
                          ELSE TB/(TB-TA) ELSE IF TB<0 THEN (TA-TB)/TA             00024530  ARC10480
                          ELSE TA/TB)                                              00024540  ARC10480
                        END;                                                       00024550  ARC10480
                       FOR I1 NFEATURES DO                                         00024560  ARC10490
                        BEGIN                                                      00024570  ARC10490
                          TB←XWATE[I]; TA←WATE[I];                                 00024580  ARC10490
                          KA←REPWATE[I];                                           00024590  ARC10490
                          IF TA=TB THEN REPWATE[I]←KA+1 ELSE                       00024600  ARC10500
                           BEGIN                                                   00024610  ARC10500
                             IF KA>10 THEN KA←10; T←1/KA;                          00024620  ARC10500
                             TC←(LV2-V2)/V2;                                       00024630  ARC10510
                             IF TC<T THEN T←TC;                                    00024640  ARC10510
                             IF T<.05 THEN T←.05;                                  00024650  ARC10510
                             XWATE[I]←T×TB+(1-T)×TA;                               00024660  ARC10520
                             KA←(KA DIV 2)+1;                                      00024670  ARC10520
                             REPWATE[I]←KA                                         00024680  ARC10520
                           END;                                                    00024690  ARC10520
                        END FOR I;                                                 00024700  ARC10530
                                                                                   00024710  ARC10531
COMMENT AT THIS POINT INITIAL ADJUSTMENTS HAVE BEEN MADE. NOW CONSISTENC           00024720  ARC10531
        Y IS CHECKED AND FINAL ADJUSTMENTS MADE;                                   00024730  ARC10532
                    TA←XWATE[1]; TB←XWATE[2];                                      00024740  ARC10540
                    TC←XWATE[3]; TD←XWATE[4];                                      00024750  ARC10540
                    IF TC<TA THEN TA←TC←(TA+TC)/2;                                 00024760  ARC10550
                    IF TB<TA THEN TA←TB←(TA+TB)/2;                                 00024770  ARC10550
                    IF TD<TC THEN TC←TD;                                           00024780  ARC10560
                    IF TD<TB THEN TB←TD; XWATE[1]←TA;                              00024790  ARC10570
                    XWATE[2]←TB; XWATE[3]←TC;                                      00024800  ARC10570
                    XWATE[4]←TD; TA←XWATE[5];                                      00024810  ARC10580
                    TB←XWATE[6]; TC←XWATE[7];                                      00024820  ARC10580
                    TD←XWATE[8];                                                   00024830  ARC10580
                    IF TA<TC THEN TA←TC←(TA+TC)/2;                                 00024840  ARC10590
                    IF TB<TA THEN TA←TB←(TA+TB)/2;                                 00024850  ARC10590
                    IF TD<TC THEN TC←TD←(TC+TD)/2;                                 00024860  ARC10600
                    XWATE[5]←TA; XWATE[6]←TB;                                      00024870  ARC10610
                    XWATE[7]←TC; XWATE[8]←TD;                                      00024880  ARC10610
                    TA←XWATE[9]; TB←XWATE[10];                                     00024890  ARC10620
                    TC←XWATE[11];                                                  00024900  ARC10620
                    IF TA<TB THEN IF TB<TC THEN TA←TB←TC←(TB+TC)/2                 00024910  ARC10630
                    ELSE TA←TB←(TA+TB)/2;                                          00024920  ARC10630
                    IF TB<TC THEN                                                  00024930  ARC10640
                     BEGIN                                                         00024940  ARC10640
                       TB←TC←(TB+TC)/2;                                            00024950  ARC10640
                       IF TA<TB THEN TA←TB;                                        00024960  ARC10640
                     END;                                                          00024970  ARC10640
                    XWATE[9]←TA; XWATE[10]←TB;                                     00024980  ARC10650
                    XWATE[11]←TC; TA←XWATE[12];                                    00024990  ARC10660
                    TB←XWATE[13];                                                  00025000  ARC10660
                    IF TB<TA THEN TA←TB←(TA+TB)/2;                                 00025010  ARC10660
```

```
                    XWATE[12]←TA; XWATE[13]←TB;                                    00025020  ARC10670
                    TC←XWATE[14]; TB←XWATE[15];                                    00025030  ARC10680
                    TA←XWATE[16]; TD←XWATE[17];                                    00025040  ARC10680
                    IF TA<TB THEN IF TB<TC THEN TA←TB←TC←(TB+TC)/2                 00025050  ARC10690
                    ELSE TB←TA←(TB+TA)/2;                                          00025060  ARC10690
                    IF TB<TC THEN                                                  00025070  ARC10700
                     BEGIN                                                         00025080  ARC10700
                       TB←TC←(TB+TC)/2;                                            00025090  ARC10700
                       IF TA<TB THEN TA←TB                                         00025100  ARC10700
                     END;                                                          00025110  ARC10700
                    IF TD<TA THEN TD←TA; XWATE[14]←TC;                             00025120  ARC10720
                    XWATE[15]←TB; XWATE[16]←TA;                                    00025130  ARC10720
                    XWATE[17]←TD; TA←XWATE[18];                                    00025140  ARC10730
                    TB←XWATE[19];                                                  00025150  ARC10730
                    IF TB<TA THEN XWATE[18]←XWATE[19]←(TA+TB)/2;                   00025160  ARC10730
                    FORI NFEATURES DO WATE[I]←XWATE[I];                            00025170  ARC10740
                    WRITE(PACDTAF[13],NFEATURES+1,REPWATE[*]);                     00025180  ARC10750
                    WRITE(PACDTAF[14],NFEATURES+1,WATE[*]);                        00025190  ARC10760
                    COMBOUT(LBESTCOMB,1);                                          00025200  ARC10770
                    WRITE(DATAOUT,<10 F10.4>,FORI NFEATURES DO                     00025210  ARC10780
                    LPFEATURE[I]); COMBOUT(BESTCOMB,2);                            00025220  ARC10790
                    WRITE(DATAOUT ,<10 F10.4>,FORI NFEATURES DO                    00025230  ARC10800
                    PFEATURE[I]); WRITE(DATAOUT[DRL]);                             00025240  ARC10801
                    WRITE(DATAOUT,<"NEW WEIGHTS",X8,8 F10.5/11 F10.                00025250  ARC10810
                    5/>,FORI1 NFEATURES DO WATE[I]);                               00025260  ARC10820
                    IF REPEAT>TREPEAT THEN                                         00025270  ARC10830
                     BEGIN                                                         00025280  ARC10830
                       TREPEAT←TREPEAT+1; REPFLAG←TRUE;                            00025290  ARC10830
                       IF V1=LV1 THEN GO TO D1                                     00025300  ARC10840
                     END                                                           00025310  ARC10840
                   END ELSE                                                        00025320  ARC10850
                   BEGIN                                                           00025330  ARC10850
                     FORI1 NFEATURES DO REPWATE[I]←REPWATE[I]+1;                   00025340  ARC10850
                     WRITE(PACDTAF[13],NFEATURES+1,REPWATE[*]);                    00025350  ARC10860
                   END IF V2;                                                      00025360  ARC10870
                 END IF ANSWER OR LEARNING;                                        00025370  ARC10880
               END DECISION ROUTINE;                                               00025380  ARC10890
                                                                                   00025390  ARC10891
COMMENT DECISION ROUTINE COMPLETE, WRITE ALLOCATION AND FEATURE VALUES;            00025400  ARC10891
               EVALCNT←EVALCNT+SUBEVALCNT;                                         00025410  ARC10900
               IF OUT7 THEN                                                        00025420  ARC10910
                BEGIN                                                              00025430  ARC10910
                  IF LEARNING OR ANSWER THEN COMBOUT(LBESTCOMB,1);                 00025440  ARC10910
                  COMBOUT(BESTCOMB,2)                                              00025450  ARC10920
                END;                                                               00025460  ARC10920
               WRITE(DATAOUT,<"NUMBER OF EVALUATIONS WAS",I10,                     00025470  ARC10930
               " TOTAL EVALUATIONS TO " ,"THIS POINT IS",I10>,                     00025480  ARC10940
               SUBEVALCNT,EVALCNT); SUBEVALCNT←0;                                  00025490  ARC10940
               WRITE(DATAOUT, </"FEATURE VALUES:     FUND   CUNDA   CT",00025500  ARC10960
FUND  CTUNDA   TLUND  TLUNDA   TLONE  TLONEA  AMZERO   AMONE   AMTWO  ",00025510  ARC10960
                DMMOB1  DMMOB2"/X16,13 F8.2,/ X16,                                 00025520  ARC10960
               "   DBALQ   DBALH   DBALT   DBALA  SOLEDM  SOLDMA" /,X16,00025530  ARC10990
               13 F8.2>,FORI1 NFEATURES DO PFEATURE[I]);                           00025540  ARC10990
               WRITE(DATAOUT,</"V1=",I4,"  V2=",F8.3,"  V3=",I4>,V1,V2,            00025550  ARC11000
               V3);                                                                00025560  ARC11000
               IF ANSWER THEN FORI FUTALT=1 DO BESTALT[I]←(IF ANSINDEX[            00025570  ARC11020
               I]≤ALTERS[1,1] THEN LBESTCOMB[I] ELSE 0);                           00025580  ARC11030
               GO TO SIMRESULT;                                                    00025590  ARC11040
SIMRESULT: COMMENT SIMULATE THE RESULT OF CURRENT PERIOD DM SELECTION;             00025600  ARC11060
                BEGIN                                                              00025610  ARC11060
LABEL             L1,L2,L3,L4,L5,L6;                                               00025620  ARC11070
```

```
                    FORI FUTALT=1 DO                                          00025630        ARC11080
                     BEGIN                                                    00025640        ARC11080
                       KF←BESTALT[I]; IF KF=0 THEN GO TO L2;                  00025650        ARC11080
                       CODE←" DM"; KA←ALTERS[I,0];                            00025660        ARC11090
                       KB←AMDATA[KA,1];                                       00025670        ARC11090
                       IF KF<34 THEN                                          00025680        ARC11090
                        BEGIN                                                 00025690        ARC11090
                          CODE←"TDM"; DLIST[KF]←DLIST[KF]-1;                  00025700        ARC11100
                          IF HEURISTIC4 OR RANDOM≤TDMPROB THEN                00025710        ARC11110
                           BEGIN                                              00025720        ARC11110
                             KG←KF;                                           00025730        ARC11110
                             GO TO L5                                         00025740        ARC11110
                           END ELSE WRITE(DATAOUT,</"TDM",I4,                 00025750        ARC11120
                          " FIRED UNSUCCESSFULLY AT AM",I4>,KF,KB);           00025760        ARC11120
                        END ELSE                                              00025770        ARC11140
                        BEGIN                                                 00025780        ARC11140
                          BREAKDMWORD(KF,KG,KG,KG,KD);                        00025790        ARC11140
                          K←SDLOCO[2×KG+KD]←SDLOCO[KG×2+KD]-1;                00025800        ARC11150
                          IF K=0 THEN SETBIT( SDMV[KD],KG);                   00025810        ARC11160
                          IF HEURISTIC4 OR RANDOM≤DMSPEC[2×KD+1] THEN GO      00025820        ARC11170
                          TO L5 ELSE WRITE(DATAOUT,</"DM",I4,I3,              00025830        ARC11180
                          " FIRED UNSUCCESSFULLY AT AM",I4>,KG,KD ,KB)        00025840        ARC11190
                        END;                                                  00025850        ARC11190
                       GO TO L2;                                              00025860        ARC11200
L5:                    WRITE(DATAOUT,KILLAM,KB,PERIOD,CODE,KG);               00025870        ARC11210
L1:                    AMDATA[KA,0]←0; KA←AMDATA[KA,7];                       00025880        ARC11220
                       IF KA≠0 THEN GO TO L1;                                 00025890        ARC11220
L2:                   END FORI LOOP;                                          00025900        ARC11230
L6:                   WRITE(DATAOUT[DBL]);                                    00025910        ARC11240
                      FORI1 MAXAMDT DO IF AMDATA[I,0]>0 THEN                  00025920        ARC11250
                       BEGIN                                                  00025930        ARC11250
                         KA←I;                                                00025940        ARC11250
                         IF AMDATA[I,2]=PERIOD THEN                           00025950        ARC11260
                          BEGIN                                               00025960        ARC11260
                            KB←AMDATA[I,1]; K←I;                              00025970        ARC11260
L3:                         AMDATA[K,0]←0; K←AMDATA[K,7];                     00025980        ARC11270
                            IF K≠0 THEN GO TO L3;                             00025990        ARC11270
L4:                         KG←ABS(AMDATA[I,4]); K←AMDATA[I,3];               00026000        ARC11280
                            T←TGTLCV[K,4]←TGTLCV[K,4]×(1-DESTPROB)            00026010        ARC11290
                            WRITE(DATAOUT,DESTTGT,KG,PERIOD,KB,T);            00026020        ARC11300
                          END;                                                00026030        ARC11300
                       END FORI LOOP;                                         00026040        ARC11320
                      IF KA<MAXAMDT THEN MAXAMDT←KA;                          00026050        ARC11320
                      FOR J←MINATT STEP 1 UNTIL NAM DO IF ATTACK[J,7]=0       00026060        ARC11330
                      THEN MINATT←J+1 ELSE J←NAM;                             00026070        ARC11340
                    END SIMRESULT PROCESS OVER;                               00026080        ARC11350
                                                                              00026090        ARC11351
COMMENT THIS COMPLETES ONE PERIOD: RETURN IS MADE TO NEXTPERIOD UNTIL         00026100        ARC11351
        ATTACK IS COMPLETE THEN JUMP IS MADE TO FINISH WHERE ALL TABLES       00026110        ARC11352
        ARE RESET TO THEIR PREATTACK VALUES AND FINAL RESULTS ARE OUTPUT      00026120        ARC11353
        IF ATTACK IS TO BE REPEATED OR A NEW ATTACK ENTERED UNDER THE         00026130        ARC11354
        LEARNING PROCESS,THIS IS DONE AND RETURN IS MADE TO NEXTPERIOD        00026140        ARC11355
        OR STARTNEWPROBLEM IF A NEW PROBLEM IS TO BE ENTERED;                 00026150        ARC11356
              GO TO NEXTPERIOD;                                               00026160        ARC11360
FULL:         WRITE(DATAOUT,<"AMDATA FULL ON AM",I4>,ATTACK[I,0]);            00026170        ARC11370
FINISH:       FINVALUE←0;                                                     00026180        ARC11390
              FORI NTGT DO FINVALUE←FINVALUE+TGTLCV[I,4];                     00026190        ARC11390
              WRITE(DATAOUT,<" "/"FINAL TARGET VALUE IS",F12.5//              00026200        ARC11400
              "FINAL FEATURE WEIGHTS",8 F10.5/11 F10.5//                      00026210        ARC11410
              "FINAL WEIGHT REPEATS   ",8 I10/11 I10//>,FINVALUE,             00026220        ARC11410
              FORI1 NFEATURES DO WATE[I],FORI1 NFEATURES DO REPWATE[I]);      00026230        ARC11420
```

```
                ; FORI1 NTGT DO TGTLOV[I,4]←SAVEINF[I];                                  00026240    ARC11430
                K←NTGT; J←NDM←NDM +1;                                                    00026250    ARC11430
                FORI1 J DO SDLOCOF[I]←SAVEINF[K←K+1]; K←NTGT+J;                          00026260    ARC11440
                FORI1 NTDLIST DO DLIST[I]←SAVEINF[K←K+1];                                00026270    ARC11450
                EVALCNT←0; SDMVF1]←SDMVE0]←0; AMFLAG←TRUE;                               00026280    ARC11460
                AMIND←0; CURATT←1; AMNOW←AMDET←MAXAMDT←0;                                00026290    ARC11460
                FOR I←2 STEP 2 UNTIL J DO FOR K ←I,I+1 DO IF SDLOCO[K]=0                 00026300    ARC11470
                THEN SETBITC SDMVEK-I],I DIV 2); PERIOD←0;                               00026310    ARC11480
                IF CUMPARUN THEN                                                         00026320    ARC11481
                 BEGIN                                                                   00026330    ARC11481
                   READ(DATAIN[NO],A,CODE)[ENDOFRUN];                                    00026340    ARC11481
                   IF CODE="WAT" THEN                                                    00026350    ARC11482
                    BEGIN                                                                00026360    ARC11482
                      READ(DATAIN,/,1,FORI1 NFEATURES DO WATE[I]);                       00026370    ARC11482
                      WRITE(DATAOUT[PAGE]);                                              00026380    ARC11483
                      WRITE(DATAOUT[ORL],<"REPEAT OF ",5 A6," USING "                    00026390    ARC11484
                      "DIFFERENT WEIGHTS">,FORI 4 DO RUNID[I] );                         00026400    ARC11484
                      WRITE(DATAOUT[ORL],< "FEATURE WEIGHTS ",8 F10.5/11                 00026410    ARC11485
                      F10.5>,FORI1 NFEATURES DO WATE[I]);                                00026420    ARC11485
                      GO TO NEXTPERIOD                                                   00026430    ARC11486
                    END                                                                  00026440    ARC11486
                 END IF COMPARUN;                                                        00026450    ARC11486
                IF CYCLE>1 THEN IF (NEWWATE AND LSAVE) OR NOT LSAVE THEN                 00026460    ARC11490
                 BEGIN                                                                   00026470    ARC11490
    COMMENT RECYCLE THE CURRENT ATTACK;                                                  00026480    ARC11500
                   NEWWATE←FALSE; WRITE(DATAOUT[PAGE]);                                  00026490    ARC11520
                   WRITE(DATAOUT,<"BEGINNING CYCLE",I3," OF ",5 A6//>,                   00026500    ARC11530
                   SAVECYCLE← CYCLE+2,FORI 4 DO RUNID[I]);                               00026510    ARC11540
                   CYCLE←CYCLE-1; BOOK2←BOOK1;                                           00026520    ARC11550
                   GO TO NEXTPERIOD;                                                     00026530    ARC11550
                 END CYCLE BACK;                                                         00026540    ARC11560
                IF BOOK1 THEN                                                            00026550    ARC11570
                 BEGIN                                                                   00026560    ARC11570
                     COMMENT CHECK FOR MORE BOOK GAMES;                                  00026570    ARC11570
LABEL            L1,L2;                                                                  00026580    ARC11580
                 BOOK2←FLAG←FALSE; WRITE(DATAOUT[PAGE]);                                 00026590    ARC11580
L1;              READ(DATAIN[NO],A,CODE)[L2];                                            00026600    ARC11590
                 IF CODE="LAS" THEN GO TO L2;                                            00026610
                 IF CODE="RUN" THEN                                                      00026620    ARC11600
                  BEGIN                                                                  00026630    ARC11600
                    READ(DATAIN,<X7,5 A6>,FORI 4 DO RUNID[I]);                           00026640    ARC11600
                    WRITE(DATAOUT,<5 A6//>,FORI 4 DO RUNID[I]);                          00026650    ARC11610
                    GO TO L1                                                             00026660    ARC11610
                  END;                                                                   00026670    ARC11610
                 IF CODE="AM " THEN                                                      00026680    ARC11620
                  BEGIN                                                                  00026690    ARC11620
                    FLAG←TRUE; GENEWATT; GO TO L1;                                       00026700    ARC11620
                  END;                                                                   00026710    ARC11620
                 IF CODE="TRA" THEN                                                      00026720    ARC11630
                  BEGIN                                                                  00026730    ARC11630
                    READ(DATAIN,/,I,I,CYCLE,I,REPEAT);                                   00026740    ARC11630
                    SAVECYCLE←CYCLE; GO TO L1                                            00026750    ARC11640
                  END;                                                                   00026760    ARC11640
                 IF CODE="END" THEN                                                      00026770    ARC11650
                  BEGIN                                                                  00026780    ARC11650
                    READ(DATAIN);                                                        00026790    ARC11650
                    IF FLAG THEN GO TO L2 ELSE GO TO L1                                  00026800    ARC11660
                  END;                                                                   00026810    ARC11660
                 READ(DATAIN); GO TO L1;                                                 00026820    ARC11670
L2;              IF FLAG THEN                                                            00026830    ARC11680
                  BEGIN                                                                  00026840    ARC11680
```

```
                FORI NTYPE=1 DO PICKCHARS(SDMV[I],DMVOID[I],1,1,8);00026850        ARC11690
                FORI NTOLIST DO TOLIST[I]+DLIST[I]; K+1;            00026860        ARC11700
                FORI1 NDM DO FOR J+3 STEP 1 UNTIL NTYPE+2 DO        00026870        ARC11710
                DMLOC[I,JJ+SOLOC0[K+K+1]; SETATTACK;                00026880        ARC11710
                GO TO NEXTPER100                                    00026890        ARC11720
              END;                                                  00026900        ARC11720
            END IFBOOK1;                                            00026910        ARC11730
          WRITE(DATAOUT,<"END OF RUN ",5 A6/>,FORI 4 DO RUNID[I]);  00026920        ARC11740
          WRITE(DATAOUT,<"FINAL RANDOM NUMBER SEED IS",I12/>,REAL(  00026930        ARC11750
          RSEED));                                                  00026940        ARC11750
          WRITE(PUNCHF,<"WATE,",6(F10.5,",")>,FORI1 6 DO WATE[I]);  00026950
          WRITE(PUNCHF,<7(F10.5,",")>,FOR I+7 STEP 1 UNTIL          00026960
          NFEATURES DO WATE[I]);                                    00026970
          WRITE(PUNCHF,<"COR,",19(I3,",")>,FORI1 NFEATURES DO       00026980
          REPWATE[I]);                                              00026990
          WRITE(PUNCHF,<40 I2>,FORI 39 DO I);                       00027000
          READ(DATAIN,A,CODE)[ENDOFRUN];                            00027010
          IF CODE#"LAS" THEN GO TO ENDOFRUN;                        00027020
          READ(DATAIN[NO],A,CODE)[ENDOFRUN];                        00027030
          GO TO STARTNEWPROBLEM;                                    00027040
        END MAIN BODY OF ARC;                                       00027050        ARC11760
ENDOFRUN:  WRITE(DATAOUT[PAGE]);                                    00027060
          WRITE(DATAOUT,<"END OF THIS PROBLEM">);                   00027070
        END.                                                        00027080        ARC11770
```

APPENDIX B

Table Descriptions and Input Formats

for Principle Tables used by ACTS.

To assist the interested reader in understanding the program, the principle tables used in ACTS are identified and the size, utilization, indexing, internal format and type (real, integer, alpha) are given, followed by the punched card format of input data if the table is generated from external information. "FREE" format on the Burroughs B5500 in general means that each entity (name or number) is followed by a comma and that association with identifiers is determined solely by relative position on the list.

DMSPEC. The range and effectiveness (probability of destroying an AM) is stored in DMSPEC for each type of area defense missile.

Table Size 0: NDMSPEC

Table Utilization 0: 2 x NTYPE - 1

Indexing I ← 2 x (ID number of desired type)

I ← I + 1

Table Format

For each type as indexed above there are two entries

Word 1: Range of DM

Word 2: Effectiveness (Probability of Success) of DM

Type Array REAL

Input Format FREE

ordered as:

SP

ID number of type

Range

Effectiveness

TDLIST. Terminal Defense Site Missile Status

Table Size 0: NTDLIST

Table Utilization 0: NTDLIST

Indexing I ← TDM site ID number

Table Format Single entry per site giving number of missiles

Type Array Integer

Input Format FREE

ordered as:

TDM

Site ID number

Number Missiles

TGTLCV. Target Location and Value

Table Size 0: MTGT , 0:4

Table Utilization 1: NTGT , 0:4

Indexing Standard

Table Format

One row per target sorted on descending word 0 . For each row:

Word 0: minimum X coordinate

Word 1: minimum Y coordinate

2: maximum X "

3: " Y "

4: value at the target

Type Array REAL

Input Format (included with TGTTAD input)

TGTTAD. Target type, identification and terminal defense sites.

Table Size 0: MTGT , 0:5

Table Utilization 1: NTGT , 0:5

Indexing Standard

Table Format

One row per target sorted as TGTLCV

For each row:

Word 0: Target Identifier

1: Target Type

2: Terminal defense site covering the target

3: " " " " " "

4: " " " " " "

5: " " " " " "

Type Array Integer

Input Format

Column	Entry
1	T
5 - 10	Target Identification
14 - 19	Target Type

23 - 25	Terminal Defense Site (right justified)
26 - 28	" " " " "
29 - 31	" " " " "
32 - 34	Terminal Defense Site (right justified)
37 - 43	Minimum X coordinate (right justified as dec. pointed)
44 - 50	Minimum Y coordinate (right justified as dec. pointed)
51 - 57	Maximum X coordinate (right justified as dec. pointed)
58 - 64	Maximum X coordinage (right justified as dec. pointed)
67 - 70	Target value

DMLOCO. DM Site Location and Missile Status

Table Size 0: MDMLOCO , 0: NTYPE + 3

Table Utilization 1: NDM , 0: NTYPE + 2

Indexing DM Site Identification Number

Table Format

One row per DM site

For each row:

Word 0: X coordinate

1: Y coordinate

2: Total missiles initially

3: Number of type 0 missiles

4: Number of type 1 missiles

Type Array REAL

Input Format FREE

Ordered as

DM

Site Identification Number

X coordinate

Y coordinate

number of type 0 missiles

number of type 1 missiles

SQRDTA. DM Sites Effecting Each Integer Grid Square

Table Size 0: NSQRDTA , 0: MSQRDTA

Table Utilization 0: NSQR , 0: NSQRDTA

Indexing I: Truncated X coordinate

J: (see format)

Table Format

One row per integer X coordinate

For each row:

Word 0: Index to DM sites effecting Y = 0 square

1: " " " " " Y = 1 "

M: " " " " " Y = M "

M+1: Last row entry + 1

M+2: DM sites effecting (0 indicates none)

(entries are indexed in 0 to N above)

Type Array Integer

Input Format (table is internally generated)

ATTACK. AM Data for an Attack

Table Size 0: MATTACK , 0: NATTACK

Table Utilization 1: NAM, 0:6

Indexing Standard

Table Format

One row per AM

For each row:

Word 0: AM Identifier

1: Time of pick up

2: X coordinate at pickup

3: Y coordinate at pickup

4: X coordinate of ground zero

5: Y coordinate of ground zero

6: Speed

Type Array REAL

Input Format FREE

Ordered as:

AM

Identifier

Time of Pick-up

X coordinate at Pick-up

Y coordinate at Pick-up

X coordinate of G. Z.

Y coordinate of G. Z.

Speed

AMDATA. Data Pertaining to Attacking Missile

Table Size 0: MAMDATA , 0: NAMDATA

Table Utilization 1: variable , 0: NAMDATA

Indexing Standard with overflow rows

Table Format

One or more rows for each AM

For each row contain -(index of previous row)

Word 0: 0 not busy, +1 busy begin, <0 busy overflow

1: AM identification

2: Time period of ground zero

3: Index to target in TGTLCV + TGTTAD

4: Target ID/Local Defense (negative if not defended)

5: Index to entry for current time group

6: Last time period in row

7: Index to overflow row

8: Index of last entry in row

9: begin AM/DM correlation in blocks of [-Time Period] followed by one word for each covering DM during the period with [period (bits 18-29), DM site (bits 30-41): Index to missile type (bits 42-47)]

Type Array Integer

Input Format (generated from AM reports)

PACDTA. Plausible Attack Corridor Data

Table Size 0: MPACDTA, 0:63

Table Utilization Row 1 4:63 , other rows 0163

Indexing Standard (see format)

Table Format

Word [3] is set to 1 thereafter groupings of 4 contiguous words for each attack

Word [1]: Bits 0-11 X coord of initial square

Bits 12-23 Y coord of initial square

Bits 24-35 X cord of GZ square

Bits 36-48 Y coord of GZ square

Word [2]: Type targets under attack--up to 8, one stored in each six bit character.

Word [3]: Type targets under attack not covered locally up to 8--one per character.

Word [4]: Last index in PACOMS for this PAC

Type Array INTEGER

Input Format FREE

Ordered as

PAC

X coord of attack square

Y coord of attack square

X coord of GZ square

Y coord of GZ square

PACDMS. DM Words Associated with the Plausible Attack Corridors

Table Size	0:255
Table Utilization	1:255
Indexing	Word [4] of each set in PACDTA gives index of last entry in PACDMS for that PAC.
Table Format	(Unformatted) packed words as in AMDATA
Type Array	Integer
Input Format	Generated internally

DMVOID. Tabulates DM Site Missile Voids by Type

Table Size	0:NTYPE
Table Utilization	0:NTYPE
Indexing	DM Type Code Number
Table Format	

One logical word per DM type. Bit I is set to 1 if DM site I is void of other DM type.

Type Array	INTEGER
Input Format	Generated internally

ALTERS. Alternate DMs covering AMs

Table Size	0:MAMDATA , 0:11
Table Utilization	(same)
Indexing	standard
Table Format	

Word 0: Index to row in AMDATA

1: Index in ALTERS of last word pertaining to current period

2: Index in ALTERS of last word in row (applying heuristics).

3: Index in ALTERS of last word in row

4: Index in ALTERS for current alternative being considered.

5: bits 12-19 ≠ periods beyond first period DM effective

(and up) : bits 18-29 period DM first effective

bits 30-41 DM site

bits 42-47 Type DM

Word 11: the DM planning group to which the AM belongs

Type Array INTEGER

Input Format Internally generated

GROUPS. Pre-specified DM site groupings to use with heuristic H6.

Table Size 0:MDMLOCO

Table Utilization 1:MDMLOCO

Indexing I ← DM site number

Table Format

Single entry per DM site giving number of the group to which the site is assigned

Type Array INTEGER

Input Format FREE

Ordered as

GROUP

(group number)

OF

(total number of groups)

(number of sites assigned to this group)

SITES

(site numbers)

APPENDIX C

Description of Principal Global Variables

Used in ACTS

Identifier	Description
NAMDATA	Number of columns in AMDATA
MAMDATA	Number of rows in AMDATA
MSQR	Largest ordinate in grid system
NSQR	Largest abscissa in grid system
MSQRDTA	Number of rows in SQRDTA
NSQRDTA	Number of columns in SQRDTA
MATTACK	Number of rows in ATTACK
NATTACK	Number of columns in ATTACK
NDMSPEC	Number of elements in DMSPEC
NTDLIST	Number of elements in TDLIST
MTGT	Number of rows in TGTLCV and TGTTAD
MDMLOCO	Number of rows in DMLOCO
MPACDTA	Number of rows in PACDTA
NPACDMS	Number of entries in PACDMS
NPAC	Number of plausible attack corridors
NDM	Number of Defense Missile Sites
NAM	Number of AM in an attack
NTGT	Number of targets in the system
NTYPE	Number of DM types identified
NTYPT	Number of target types identified
AMDET	Number of AM detected since attack began

Identifier	Description
AMNOW	Number of AM currently in the system
ESTOTAM	Number of AM estimated in possession of the enemy
TYPATK	Logical word-estimate of type of attack
PERIOD	Current time period
TWOCOR	Logical - Bit N set if DM site N is included in DMMOB1
THREECOR	Logical - Bit N set if DM site N is included in DMMOB2
HEURISTIC 1	Boolean - true if heuristic H1 is active
HEURISTIC 2	Boolean - true if heuristic H2 is active
HEURISTIC 3	Boolean - true if heuristic H3 is active
HEURISTIC 4	Boolean - true if heuristic H4 is active
HEURISTIC 5	Boolean - true if heuristic H5 is active
HEURISTIC 6	Boolean - true if heuristic H6 is active
HEURISTIC 7	Boolean - true if heuristic H7 is active
HEURISTIC 8	Boolean - true if heuristic H8 is active
HEURISTIC 9	Boolean - true if heuristic H9 is active
SQOK	Boolean - true if SQRDTA has been built
REPFLAG	Boolean - true if an attack is to be repeated
LEARNING	Boolean - true if training run is being made
BOOK1	Boolean - true if key allocation specified this attack
ANSWER	Boolean - true if key allocation specified this period
ANSWAS	Boolean - true if key allocation found this period
COMPARUN	Boolean - true if attack is to be repeated using differing feature weights
ADJWATE	Boolean - true if new weights determined are to be saved in permanent file

Identifier	Description
NEWATE	Boolean - true if weights are adjusted
NODEFENSE	Boolean - true if there is no defense against an AM in the system
NOTGT	Boolean - true if an AM will not strike a target
TOOLATE	Boolean - true if AM detection is too late to provide a defense
NFEATURES	Number of features in the decision function
TDMPROB	Probability of kill of a Local Defense Missile
DESTPROB	Probability of destruction of a target by a successful AM
TOTVALUE	Sum total value of targets before attack
FINVALUE	Sum total value of targets after attack
REPEAT	Number of times an attack is repeated for learning
CYCLE	Number of times a situation is to be cycled seeking sufficient weight adjustment to yield the key allocation
MINTGTVAL	Minimum target value under heuristic H8
ACTIVEINDEX	Number of periods during which a type target is not attack before being considered inactive
NUMGRPS	Number of planning groups in the current period

file:
MOLGEN

APPENDIX D

Summary of Attacks

The training and comparison attacks referenced in Chapters V and VI are listed below. Following the attacking missile number is the location and time of detection and the identification and location of the target. Beneath each listing the area (DM) and local (TDM) missile sites relevant to the attack are identified with the number of missiles assigned for the attack.

TRAINING ATTACK TA1 MUBILITY AND BALANCE

AM	DETECTION X	Y	TIME	TARGET IDENT	X	Y
0	19.00	9.00	1.00	D2	11.80	14.06
1	19.00	9.00	1.00	D2	11.80	14.06
2	19.00	9.50	2.00	P4	8.60	14.85
3	19.00	9.00	2.00	D2	11.80	14.06
4	19.00	9.50	3.00	P4	8.60	14.85
5	19.00	9.00	3.00	D2	11.80	14.06
6	19.00	9.50	4.00	P4	8.60	14.85
7	19.00	9.00	4.00	D2	11.80	14.06
8	19.00	9.50	5.00	P4	8.60	14.85
9	19.00	9.00	5.00	D2	11.80	14.06
10	19.00	9.50	6.00	P4	8.60	14.85
11	19.00	9.00	6.00	D2	11.80	14.06
12	19.00	9.50	7.00	P4	8.60	14.85
13	19.00	9.00	7.00	D2	11.80	14.06

DM SITE	4	5	6	7	8	9
NUMB DM	4	4	4	5	2	3

TDM SITE	1	2	3	7	20	22
NUMB TDM	2	2	2	2	2	2

TABLE D1

TRAINING ATTACK TA2 PAC COVERAGE

AM	DETECTION X	DETECTION Y	TIME	TARGET IDENT	TARGET X	TARGET Y
1	19.00	9.00	1.00	D2	11.80	14.06
2	19.00	10.00	1.00	P4	8.60	14.85
3	17.00	13.00	2.00	P4	8.60	14.85
4	19.00	9.00	2.00	D2	11.80	14.06
5	19.00	10.00	3.00	P4	8.60	14.85
6	17.00	13.00	3.00	P4	8.60	14.85
7	19.00	9.00	4.00	D2	11.80	14.06
8	19.00	10.00	4.00	P4	8.60	14.85
9	17.00	13.00	5.00	P4	8.60	14.85
10	19.00	9.00	5.00	D2	11.80	14.06
11	19.00	10.00	6.00	P4	8.60	14.85
12	17.00	13.00	6.00	P4	8.60	14.85
13	19.00	9.00	7.00	D2	11.80	14.06
14	19.00	10.00	7.00	P4	8.60	14.85
15	17.00	13.00	8.00	P4	8.60	14.85
16	19.00	9.00	8.00	D2	11.80	14.06

DM SITE	4	5	6	7	8	9
NUMB DM	4	4	4	5	2	3

TDM SITE	1	2	3	7	20	22
NUMB TDM	2	2	2	2	2	2

TABLE D2

TRAINING ATTACK TA3 EXPECTED LOSS TRADEOFF

	DETECTION			TARGET		
AM	X	Y	TIME	IDENT	X	Y
1	19.00	9.00	1.00	D2	11.80	14.06
2	19.00	10.00	1.00	P4	8.60	14.85
31	19.00	7.00	1.00	PI2	11.45	14.90
3	17.00	13.00	2.00	P4	8.60	14.85
4	19.00	9.00	2.00	D2	11.80	14.06
32	19.00	7.00	2.00	P4	8.60	14.85
5	19.00	10.00	3.00	P4	8.60	14.85
6	17.00	13.00	3.00	P4	8.60	14.85
33	19.00	7.00	3.00	PI2	11.45	14.90
7	19.00	9.00	4.00	D2	11.80	14.06
8	19.00	10.00	4.00	P4	8.60	14.85
34	19.00	7.00	4.00	P4	8.60	14.85
9	17.00	13.00	5.00	P4	8.60	14.85
10	19.00	9.00	5.00	D2	11.80	14.06
35	19.00	7.00	5.00	PI2	11.45	14.90
11	19.00	10.00	6.00	P4	8.60	14.85
12	17.00	13.00	6.00	P4	8.60	14.85
36	19.00	7.00	6.00	P4	8.60	14.85
13	19.00	9.00	7.00	D2	11.80	14.06
14	19.00	10.00	7.00	P4	8.60	14.85
37	19.00	7.00	7.00	PI2	11.45	14.90
15	17.00	13.00	8.00	P4	8.60	14.85
16	19.00	9.00	8.00	D2	11.80	14.06

DM SITE	4	5	6	7	8	9
NUMB DM	4	4	4	5	2	3

TDM SITE	1	2	3	7	20	22
NUMB TDM	2	2	2	2	2	2

TABLE D3

TRAINING ATTACK TA4 TERMINAL ATTACK VIA PAC

	DETECTION			TARGET		
AM	X	Y	TIME	IDENT	X	Y
101	19.00	10.00	1.00	S1	8.80	14.80
102	19.00	10.00	1.00	D2	11.80	14.06
103	19.00	3.00	1.00	S7	6.35	11.65
104	19.00	4.00	2.00	S4	4.40	14.60
105	19.00	7.00	2.00	S4	4.40	14.60
106	13.00	14.00	2.00	D3	6.75	11.77
107	18.00	13.00	2.00	S1	8.80	14.80
108	18.00	13.00	3.00	D1	11.65	15.00
109	19.00	6.00	3.00	D3	6.75	11.77
110	13.00	13.00	3.00	S2	5.20	14.36
111	19.00	5.00	4.00	P7	2.20	13.60
112	19.00	11.00	4.00	S1	8.80	14.80
113	19.00	1.00	5.00	S7	6.35	11.65
114	19.00	2.00	5.00	P4	8.60	14.85
115	19.00	8.00	5.00	D2	11.80	14.06
116	19.00	8.00	5.00	D2	11.80	14.06
117	19.00	9.00	6.00	D1	11.65	15.00
118	18.00	13.00	6.00	S1	8.80	14.80
119	18.00	13.00	6.00	PI2	11.45	14.90
120	19.00	7.00	7.00	D1	11.65	15.00
121	19.00	8.00	7.00	D2	11.80	14.06

DM SITE	1	2	3	4	5	6	7	8	9	10	11	13
NUMB DM	2	2	4	4	4	4	5	2	1	3	2	1

TDM SITE	1	2	3	7	20	22
NUMB TDM	2	2	2	2	2	2

TABLE D4

TRAINING ATTACK TA5 TERMINAL ATTACK NOT VIA PAC

AM	DETECTION X	DETECTION Y	TIME	TARGET IDENT	TARGET X	TARGET Y
130	19.00	3.00	1.00	S7	6.35	11.65
131	19.00	4.00	1.00	S1	8.80	14.80
132	19.00	3.00	2.00	S10	5.89	10.79
133	19.00	1.00	2.00	S1	8.80	14.80
134	19.00	1.00	2.00	S4	4.40	14.60
135	19.00	9.00	2.00	S6	3.80	14.20
136	19.00	7.00	3.00	P4	8.60	14.85
137	19.00	8.00	3.00	PI1	11.40	16.20
138	19.00	10.00	3.00	D1	11.65	15.00
139	19.00	11.00	4.00	D1	11.65	15.00
140	19.00	12.00	4.00	D2	11.80	14.06
141	19.00	14.00	5.00	S1	8.80	14.80
142	18.00	14.00	5.00	D2	11.80	14.06
143	14.00	14.00	5.00	S2	5.20	14.36
144	18.00	12.30	5.00	D3	6.75	11.77
145	19.00	7.00	6.00	P1	11.65	15.20
146	19.00	10.00	6.00	P11	12.30	8.50
147	19.00	11.00	6.00	P4	8.60	14.85
148	19.00	1.00	7.00	D2	11.80	14.06
149	19.00	10.00	7.00	S5	4.20	14.40

DM SITE	1	2	3	4	5	6	7	8	9	10	11	13
NUMB DM	2	2	4	4	4	4	5	2	3	3	2	1

TDM SITE	1	2	3	7	20	22
NUMB TDM	2	2	2	2	2	2

TABLE D5

TRAINING ATTACK TA6 INTERMEDIATE ATTACK PHASE

AM	DETECTION X	DETECTION Y	TIME	TARGET IDENT	TARGET X	TARGET Y
170	18.00	14.00	1.00	S1	8.80	14.80
171	18.00	14.00	1.00	S7	6.35	11.65
172	19.00	12.00	2.00	D1	11.65	15.00
173	19.00	9.00	2.00	D2	11.80	14.06
174	19.00	12.00	2.00	D3	6.75	11.77
175	19.00	7.00	3.00	S1	8.80	14.80
176	19.00	9.00	3.00	S9	6.30	10.62
177	18.00	14.00	3.00	D2	11.80	14.06
178	19.00	7.00	4.00	S7	6.35	11.65
179	19.00	4.00	4.00	PI2	11.45	14.00
180	19.00	1.00	5.00	D3	6.75	11.77
181	19.00	4.00	5.00	S1	8.80	14.80
182	19.00	4.00	5.00	D1	11.65	15.00
183	18.00	13.00	6.00	P4	8.60	14.85
184	19.00	5.00	6.00	P6	6.85	11.93
185	19.00	9.00	7.00	D1	11.65	15.00
186	19.00	10.00	7.00	S1	8.80	14.80
187	19.00	7.00	7.00	S8	6.70	10.52
188	19.00	9.00	8.00	P3	11.80	13.50
189	19.00	10.00	8.00	P2	11.90	14.40
190	17.30	11.00	9.00	S7	6.35	11.65
191	18.00	13.00	10.00	P4	8.60	14.85
192	19.00	8.00	10.00	D1	11.65	15.00

DM SITE	1	2	3	4	5	6	7	8	9	10	11	13
NUMB DM	2	2	4	4	4	4	5	2	2	3	2	1

TDM SITE	1	2	3	7	20	22
NUMB TDM	2	2	2	2	2	2

TABLE D6

COMPARISON RUN 1 PLAUSIBLE CORRIDOR ATTACK

AM	DETECTION X	DETECTION Y	TIME	TARGET IDENT	TARGET X	TARGET Y
1	14.00	15.00	1.00	PI2	11.45	14.90
2	13.00	13.00	1.00	P6	6.85	11.93
3	19.00	9.00	1.00	PI2	11.45	14.90
4	19.00	8.00	1.00	P8	3.40	9.60
5	19.00	6.00	1.00	P4	8.60	14.85
6	19.00	4.00	1.00	M1	3.50	9.80
7	13.50	13.00	2.00	S1	8.80	14.80
8	19.00	11.00	2.00	P1	11.65	15.20
9	19.00	10.00	2.00	P4	8.60	14.85
10	19.00	7.00	2.00	S2	5.20	14.36
11	19.00	5.00	2.00	P6	6.85	11.93
12	19.00	4.00	2.00	S1	8.80	14.80
13	13.50	13.00	3.00	S3	4.89	14.37
14	19.00	10.00	3.00	P4	8.60	14.85
15	19.00	8.00	3.00	S7	6.35	11.65
16	19.00	7.00	3.00	P2	11.90	14.40
17	19.00	4.00	3.00	S4	4.40	14.60
18	19.00	3.00	3.00	P8	3.40	9.60
19	14.00	15.00	4.00	PI2	11.45	14.90
20	13.00	13.00	4.00	P6	6.85	11.93
21	19.00	9.00	4.00	PI2	11.45	14.90
22	19.00	8.00	4.00	P8	3.40	9.60
23	19.00	6.00	4.00	P4	8.60	14.85
24	19.00	4.00	4.00	M1	3.50	9.80
25	13.50	13.00	5.00	S1	8.80	14.80
26	19.00	11.00	5.00	P1	11.65	15.20
27	19.00	10.00	5.00	P4	8.60	14.85
28	19.00	7.00	5.00	S6	3.80	14.20
29	19.00	5.00	5.00	P6	6.85	11.93
30	19.00	4.00	5.00	S1	8.80	14.80
31	13.50	13.00	6.00	S5	4.20	14.40
32	19.00	10.00	6.00	P4	8.60	14.85
33	19.00	8.00	6.00	S7	6.35	11.65
34	19.00	7.00	6.00	P2	11.90	14.40
35	19.00	4.00	6.00	S4	4.40	14.60
36	19.00	3.00	7.00	P8	3.40	9.60
37	14.00	15.00	7.00	PI2	11.45	14.90
38	13.00	13.00	7.00	P6	6.85	11.93
39	19.00	9.00	7.00	PI2	11.45	14.90

TABLE D7 (a)

COMPARISON RUN 1 PLAUSIBLE CORRIDOR ATTACK
CONTINUED

	DETECTION			TARGET		
AM	X	Y	TIME	IDENT	X	Y
40	19.00	8.00	7.00	P8	3.40	9.60
41	19.00	6.00	8.00	P4	8.60	14.85
42	19.00	4.00	8.00	M1	3.50	9.80
43	13.50	13.00	8.00	S1	8.80	14.80
44	19.00	11.00	8.00	P1	11.65	15.20
45	19.00	10.00	8.00	P4	8.60	14.85
46	19.00	7.00	9.00	S6	3.80	14.20
47	19.00	5.00	9.00	P6	6.85	11.93
48	19.00	4.00	9.00	S1	8.80	14.80
49	13.50	13.00	9.00	S5	4.20	14.40
50	19.00	10.00	9.00	P4	8.60	14.85
51	19.00	8.00	10.00	S7	6.35	11.65
52	19.00	7.00	10.00	P2	11.90	14.40
53	19.00	4.00	10.00	S4	4.40	14.60
54	19.00	3.00	10.00	I2	3.50	10.10
55	14.00	15.00	10.00	P3	11.80	13.50
56	19.00	5.00	10.00	P6	6.85	11.93

DM SITE	2	3	4	5	6	7	8	9	10	11	12	13
NUMB DM	4	5	4	5	4	4	4	5	4	4	4	3

TDM SITE	1	2	3	7	20	21	22
NUMB TDM	2	2	2	1	2	2	3

TABLE D7 (b)

COMPARISON RUN 2 S,D,M TYPE TARGET ATTACK

AM	DETECTION X	DETECTION Y	TIME	TARGET IDENT	TARGET X	TARGET Y
1	2.80	12.00	0.00	S11	8.10	4.55
2	8.70	15.00	0.00	DS1	10.40	9.70
3	13.10	15.00	0.00	DS2	11.40	11.20
4	2.80	12.00	0.40	DS1	10.40	9.70
5	8.60	15.00	0.40	S7	6.35	11.65
6	13.70	15.00	0.50	S1	8.80	14.80
7	20.00	7.00	0.80	S2	5.20	14.36
8	19.00	5.00	0.90	S3	4.89	14.37
9	13.50	15.50	0.90	DS2	11.40	11.20
10	19.50	6.00	1.10	S5	4.20	14.40
11	19.00	2.00	1.10	S6	3.80	14.20
12	19.00	4.50	1.20	S4	4.40	14.60
13	20.00	10.00	1.20	S8	6.70	10.52
14	19.50	2.50	1.50	S9	6.30	10.62
15	19.00	3.00	1.50	S10	5.89	10.79
16	19.00	2.50	1.70	S11	8.10	4.55
17	20.00	6.00	1.70	S2	5.20	14.36
18	19.30	11.00	1.90	S1	8.80	14.80
19	19.00	4.80	2.50	S3	4.89	14.37
20	19.00	4.00	2.50	S4	4.40	14.60
21	19.00	3.00	2.50	M5	8.70	4.60
22	18.00	5.50	2.50	S6	3.80	14.20
23	20.00	10.00	2.70	S7	6.35	11.65
24	19.00	3.00	2.70	D3	6.75	11.77
25	19.00	4.00	3.00	DS1	10.40	9.70
26	19.00	5.00	3.10	S10	5.89	10.79
27	19.00	12.00	3.40	S11	8.10	4.55
28	13.00	15.00	3.70	DS1	10.40	9.70
29	9.00	15.00	3.70	DS2	11.40	11.20
30	17.00	13.50	4.00	D1	11.65	15.00
31	9.00	15.00	4.20	S1	8.80	14.80
32	13.50	15.50	4.20	D2	11.80	14.06
33	8.50	15.00	4.60	P11	12.30	8.50
34	20.00	6.00	4.80	S2	5.20	14.36
35	19.00	4.80	4.90	S3	4.89	14.37
36	19.00	4.00	5.10	S4	4.40	14.60
37	19.50	8.50	5.40	M1	3.50	9.80
38	18.00	5.50	5.40	S6	3.80	14.20
39	20.00	10.00	5.40	S8	6.70	10.52

TABLE D8 (a)

COMPARISON RUN 2 S,D,M TYPE TARGET ATTACK
CONTINUED

	DETECTION			TARGET		
AM	X	Y	TIME	IDENT	X	Y
40	19.00	2.50	5.50	S9	6.30	10.62
41	8.50	15.00	5.70	D5	11.30	8.65
42	19.00	5.00	5.70	S10	5.89	10.79
43	19.00	12.00	6.00	S11	8.10	4.55
44	19.70	2.60	6.00	D3	6.75	11.77
45	13.10	14.00	6.50	D3	6.75	11.77
46	20.00	8.00	6.50	DS2	11.40	11.20
47	19.50	11.50	6.50	D1	11.65	15.00
48	9.00	15.20	6.60	D2	11.80	14.06
49	19.00	11.50	6.90	S7	6.35	11.65
50	17.00	14.80	7.00	D5	11.30	8.65
51	8.30	15.00	7.40	D6	7.95	5.05
52	19.50	11.50	7.40	D6	7.95	5.05
53	20.00	9.00	7.40	D6	7.95	5.05
54	12.00	14.50	7.70	NONE	0.00	0.00
55	18.00	11.50	7.80	DS1	10.40	9.70
56	18.00	2.70	8.00	M8	8.30	3.70
57	20.00	7.00	8.00	M5	8.70	4.60
58	19.00	3.50	8.50	M1	3.50	9.80
59	8.50	15.00	8.60	M2	12.25	7.84
60	20.00	8.00	8.70	M3	12.35	7.00
61	12.00	14.00	8.80	M4	10.75	8.85
62	20.00	5.00	9.00	S5	4.20	14.40
63	19.00	4.70	9.20	M6	8.35	4.50
64	20.00	7.00	9.40	M7	7.70	7.20
65	19.00	2.40	9.50	M8	8.30	3.70
66	19.00	4.00	9.70	M9	7.40	4.80
67	19.00	2.50	9.90	M10	7.10	4.50
68	19.00	3.00	10.10	M11	6.50	3.60
69	19.00	4.90	10.30	M12	5.10	4.15
70	19.00	4.50	10.50	M13	4.75	3.75
71	19.00	0.60	10.80	M14	4.40	3.20
72	20.00	8.00	11.00	M3	12.35	7.00
73	19.00	11.00	11.20	D1	11.65	15.00
74	19.00	11.00	11.30	D2	11.80	14.06
75	19.00	3.00	11.50	D3	6.75	11.77
76	19.00	9.00	11.60	S1	8.80	14.80
77	11.30	12.00	12.00	D4	12.23	8.40
78	19.00	4.00	12.00	S2	5.20	14.36

TABLE D8 (b)

COMPARISON RUN 2 S,D,M TYPE TARGET ATTACK
CONTINUED

AM	DETECTION X	DETECTION Y	TIME	TARGET IDENT	TARGET X	TARGET Y
79	19.00	11.50	12.20	D5	11.30	8.65
80	18.00	6.90	12.40	S3	4.89	14.37
81	19.00	7.50	12.60	D6	7.95	5.05
82	8.30	15.00	12.60	D7	8.80	5.50
83	19.00	2.30	12.90	S7	6.35	11.65
84	18.50	11.50	13.10	S9	6.30	10.62
85	18.00	11.50	13.70	S11	8.10	4.55
86	11.50	13.50	14.00	D7	8.80	5.50
87	19.00	2.00	14.50	M8	8.30	3.70
88	19.00	9.00	14.50	M10	7.10	4.50
89	19.00	9.00	14.50	M11	6.50	3.60
90	19.00	9.00	14.60	M12	5.10	4.15
91	19.00	8.00	14.90	M14	4.40	3.20
92	17.00	14.00	15.10	D7	8.80	5.50
93	19.50	0.50	15.30	M1	3.50	9.80
94	13.00	14.00	15.30	M3	12.35	7.00
95	19.00	3.00	15.60	M6	8.35	4.50
96	19.00	4.00	15.90	M8	8.30	3.70
97	18.00	11.00	16.10	M2	12.25	7.84
98	18.00	11.00	16.90	M3	12.35	7.00
99	19.00	9.00	17.00	M4	10.75	8.85

DM SITE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
NUMB DM	8	8	8	6	5	6	5	5	5	5	8	5	8	5	8	5	8	8	5

TDM SITE	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
NUMB TDM	3	5	3	3	3	5	3	5	5	5	3	3	3	6	6	9	3	2	2	3	3	3

TABLE D8 (c)

APPENDIX E

Basic Target and Missile Site Date

The following tables provide detailed information about the targets in the test environment depicted in Figure 5.1. For each target, the type resources identified with the target, the defending local defense sites, minimum and maximum grid coordinates defining the boundaries of the target and the target value are listed. For each area defense missile site the grid coordinates are given. Number of DM assigned to each site is specified with the attack script.

LIST OF TARGETS USED IN TEST ENVIRONMENT

IDENT	TYPE	COVERING TDM SITES	MINIMUM COORDS X	MINIMUM COORDS Y	MAXIMUM COORDS X	MAXIMUM COORDS Y	VALUE
PI1	PI	0	16.00	11.40	16.20	11.60	10
P1	P	1	15.10	11.65	15.20	11.80	10
D1	D	1	14.95	11.65	15.00	11.70	10
S1	S	22	14.70	8.80	14.80	8.90	10
P4	P	22	14.67	8.60	14.85	8.80	10
PI2	PI	1, 7	14.60	11.45	14.90	11.70	10
S4	S	0	14.50	4.40	14.60	4.41	10
S5	S	0	14.30	4.20	14.40	4.21	10
S3	S	0	14.36	4.89	14.37	4.90	10
S2	S	0	14.35	5.20	14.36	5.21	10
P2	P	2	14.20	11.90	14.40	12.10	10
S6	S	0	14.19	3.80	14.20	3.81	10
I1	I	0	14.00	9.90	14.10	10.00	5
D2	D	2	14.00	11.80	14.06	11.90	10
P5	P	0	13.70	4.80	13.80	4.90	10
P7	P	0	13.50	2.20	13.60	2.30	10
P3	P	0	13.30	11.80	13.50	12.00	10
P6	P	3	11.80	6.85	11.93	7.00	10
D3	D	3	11.70	6.75	11.77	6.80	10
S7	S	20	11.60	6.35	11.65	6.40	10
DS2	DS	13	11.10	11.40	11.20	11.50	10
S10	S	0	10.78	5.89	10.79	5.90	10
S9	S	0	10.61	6.30	10.62	6.31	10
S8	S	0	10.51	6.70	10.52	6.71	10
I2	I	21	9.90	3.50	10.10	3.60	5
M1	M	21	9.70	3.50	9.80	3.60	5
DS1	DS	12	9.60	10.40	9.70	10.50	10
P8	P	21	9.40	3.40	9.60	3.60	10
P9	P	0	9.00	8.85	9.20	9.00	10
M4	M	11	8.80	10.75	8.85	10.80	5
D5	D	5	8.60	11.30	8.65	11.35	10
P12	P	11	8.50	10.60	8.70	10.80	10
PI7	PI	0	8.40	11.90	8.50	12.10	10
D4	D	4	8.34	12.23	8.40	12.28	10
P11	P	4, 8	8.20	12.30	8.50	12.40	10

TABLE E1 (a)

LIST OF TARGETS USED IN TEST ENVIRONMENT

IDENT	TYPE	COVERING TDM SITES	MINIMUM COORDS X	MINIMUM COORDS Y	MAXIMUM COORDS X	MAXIMUM COORDS Y	VALUE
PI6	PI	10	8.10	11.80	8.30	12.00	10
P10	P	8, 4	8.00	12.30	8.20	12.50	10
P19	P	10	7.90	11.90	8.10	12.10	10
M2	M	8, 9	7.80	12.25	7.84	12.30	5
PI3	PI	8, 9	7.80	12.30	8.00	12.50	10
PI5	PI	9, 10	7.70	12.00	7.90	12.20	10
PI4	PI	9	7.60	12.20	7.80	12.40	10
I3	I	0	7.10	7.50	7.10	7.50	6
M7	M	0	7.10	7.70	7.20	7.80	5
M3	M	0	6.80	12.35	7.00	12.50	5
P17	P	0	6.00	8.60	6.20	8.80	10
D7	D	6	5.40	8.80	5.50	8.90	10
D6	D	17	4.95	7.95	5.05	8.03	10
PI8	PI	14	4.80	8.70	5.00	8.90	10
P13	P	14, 16	4.80	8.50	5.00	8.70	10
M9	M	0	4.70	7.40	4.80	7.50	5
P15	P	16	4.70	8.30	4.90	8.50	10
PI9	PI	15	4.60	8.70	4.80	8.90	10
PI11	PI	16	4.60	8.10	4.80	8.30	10
P14	P	14, 15	4.60	8.50	4.80	8.70	10
S11	S	16	4.50	8.10	4.55	8.15	10
P16	P	16	4.50	8.30	4.70	8.50	10
M5	M	15	4.40	8.70	4.60	8.80	5
M6	M	16	4.40	8.35	4.50	8.40	5
M10	M	0	4.40	7.10	4.50	7.20	5
PI10	PI	15	4.40	8.50	4.60	8.70	10
M12	M	19	4.10	5.10	4.15	5.15	5
P18	P	19	3.80	4.80	4.00	5.00	10
PI12	PI	18	3.70	8.20	3.90	8.40	10
M13	M	19	3.70	4.75	3.75	4.80	5
M8	M	18	3.60	8.30	3.70	8.50	5
M11	M	0	3.50	6.50	3.60	6.60	5
M14	M	0	3.10	4.40	3.20	4.50	5

TABLE E1 (b)

AREA DEFENSE MISSILE SITE LOCATIONS

(RANGE OF MISSILES IS 1 SQUARE LENGTH. QUANTITY OF MISSILES AT EACH SITE IS ESTABLISHED IN THE ATTACK SCRIPT AND ARE LISTED IN APPENDIX D.)

SITE	ABSCISSA	ORDINATE
1	17.00	2.00
2	17.00	4.00
3	17.00	6.00
4	18.00	8.00
5	18.00	10.00
8	15.00	13.00
7	16.20	10.30
6	16.90	12.40
9	13.70	12.00
10	14.00	6.40
11	14.20	4.60
12	11.60	3.60
13	11.50	9.00
14	10.00	13.00
15	8.60	13.50
16	6.60	13.00
17	5.40	11.00
18	3.80	10.00
19	4.00	7.90

TABLE E2

APPENDIX F

Proof of Weight Adjustment Lemma

Under the definitions of W, D, F, F' and $\bar{W}$ given in section 4.1.2 the following holds.

Lemma: For any fixed, finite W, D, F and F', there exists a finite scalar $S^* \geq 0$ such that for any $S > S^*$, $\bar{W}F' < \bar{W}F$.

Proof:

$\infty > WF' - WF \geq 0$ by (i) section 4.1.2

$\infty > DF - DF' > 0$ by (iv) section 4.1.2.

Choose

(v) $S^* = \frac{WF'-WF}{DF-DF'}$ so that $\infty > S^* \geq 0$

and

$$S^* (DF-DF') = WF' - WF .$$

Now, for any $S > S^*$, say S' ,

$$S'(DF-DF') > WF' - WF$$

hence

$$S'DF > WF' - WF + S'DF'$$

Now,

$$\bar{W}F' = (W + S'D)F' = WF' + S'DF'$$

$$= WF' + WF - WF + S'DF'$$

$$= WF + (WF' - WF + S'DF')$$

$$< WF + S'DF = \bar{W}F .$$

The proof is complete.