[22] Tony Leal

Technical Report PTR-1024-75-11
November, 1975

ADAPTIVE DECISIONS IN BALLISTIC MISSILE DEFENSE

ANTONIO LEAL

Principal Investigator: Dr. Amos Freedy
Subcontract: TRW No A62201/LJBE
Contract: DASG-60-75-C-0084

PERCEPTRONICS

6271 VARIEL AVENUE • WOODLAND HILLS • CALIFORNIA 91364 • PHONE (213) 884-7470

TECHNICAL REPORT PTR-1024-75-11
November, 1975

ADAPTIVE DECISIONS IN BALLISTIC MISSILE DEFENSE

ANTONIO LEAL

Principal Investigator: Dr. Amos Freedy
Subcontract: TRW No. A62201/LJBE
Contract: DASG-60-75-C-0084

PERCEPTRONICS
6271 VARIEL AVENUE • WOODLAND HILLS • CALIFORNIA 91364 • PHONE (213) 884-7470

TABLE OF CONTENTS

FOREWORD

The study approach, analysis, and results described in this report were conducted by Perceptronics under subcontract to TRW Systems Group, Redondo Beach. The effort formed part of Contract No. DASG60-76-C-0084 with the U.S. Army Ballistic Missile Defense Advanced Technology Center (BMDATC), Huntsville, Alabama. The TRW Principal Investigator was Mr. John Chu. Many useful contributions were amde by other TRW personnel including Dr. F.G. Spadaro, Mr. Hans Kaspar, and Mr. Eliot Bailis. In addition, many of the adaptive programming techniques utilized in the present study were initially developed, in related form, under Contract No. N00014-73-C00286, "Adaptive Computer Aiding in Dynamic Decision Processes", which was monitored by the ONR Engineering Psychology Programs, and supported by the ARPA Human Resources Research Office.

1. INTRODUCTION

Computer techniques taken from the fields of artificial intelligence and adaptive programming have been successfully used to solve practical problems in the area of Ballistic Missile Defense. The techniques were applied to the important case of allocating interceptor missiles to an attack by enemy re-entry missiles against ICBM silos. Simulation results (see Figure 1) showed that a silo installation could be acceptably defended by only about 60% the number of interceptors required under the "taper-linear defense" strategy. This significant savings translates directly into security, because interceptors remain to defend against subsequent strikes, and into dollars, because interceptors account for a great portion of defense costs. A detailed explantion of how these results were obtained is presented in this report.

The approach centers on the method of heuristic search, which is a technique that allows analysis and evaluation of consequences arising from possible future situations. These situations are simulated by testing known actions and decisions against the present state of affairs. Then, by comparing the results, a best action can be chosen. Repetitive application of this technique yields an action sequence that determines a resource allocation. The process is termed "search" because all possible future situations resulting from available actions are generated within the constraints of the problem domain and the existing computer resources. Evaluating and comparing them corresponds to a search for a desirable end and an identification of the strategies needed to attain that end. Since all possible attack strategies and defense allocations are considered, there is no attack strategy for which the defense is unprepared. The process is termed "heuristic" because central to the analysis is a mechanism for evaluating the worth of generated future situations. For example, some of the important heuristic parameters basic for interceptor planning are

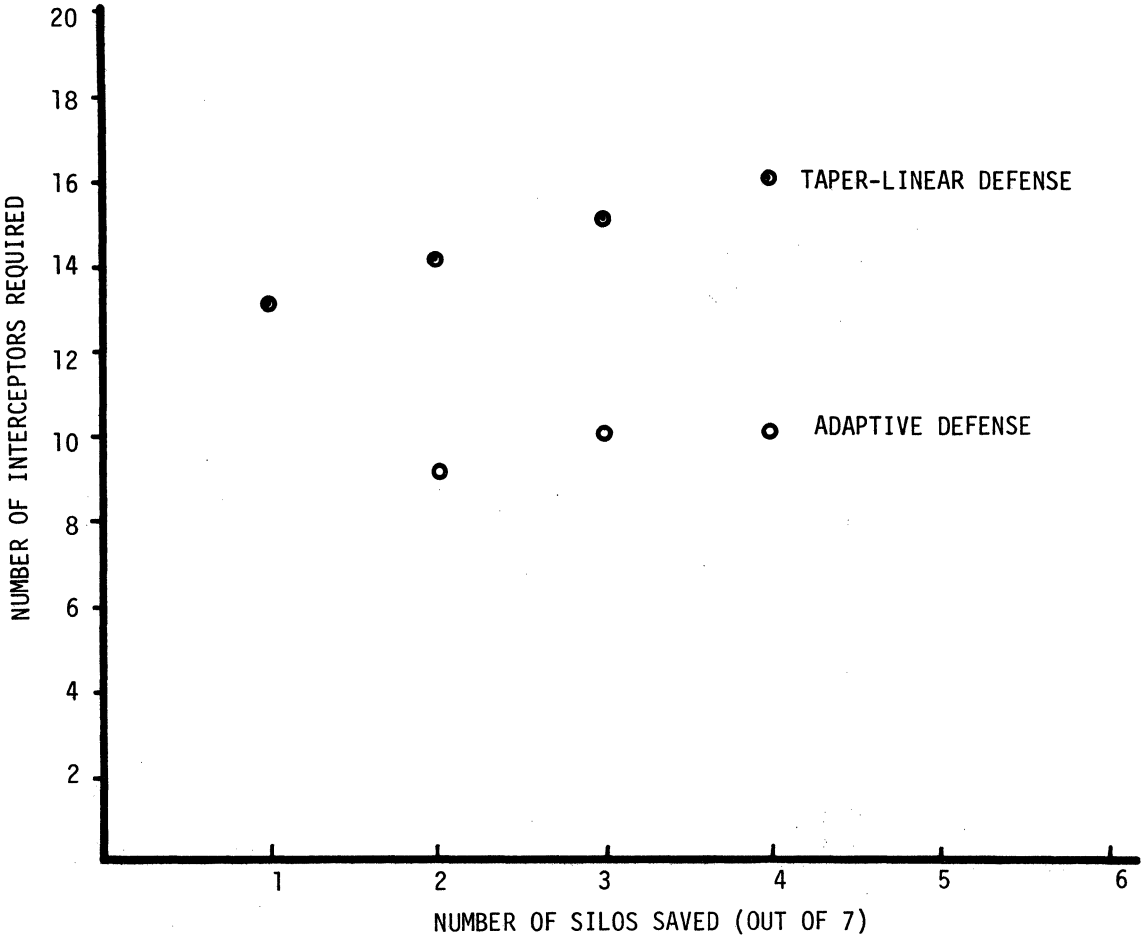


FIGURE 1. INTERCEPTOR MISSILE ALLOCATION SIMULATION RESULTS

PERCEPTRONICS

(1) the number of remaining silos, (2) the number of remaining interceptors, (3) the expected number of remaining re-entry vehicles, (4) the current interceptor coverage, etc. These and other parameters are combined to form the evaluation algorithm.

One of the most important advantages of these types of logical processes is their ability to learn from experience and adapt to a changing environment. The defensive strategy is responsive to the current threat and is not fixed throughout the engagement. Thus, there is no necessity to force security upon a predetermined strategy. The best strategy is formulated as the battle is taking place.

From the view point of Decision Analysis, solving problems by heuristic search corresponds to the real-time construction of decision structures for the purpose of selecting a best course of action from a number of available alternatives. However, the available alternatives and predicted consequences are generated and evaluated automatically, thus relieving the decision maker from complex and time-consuming analysis before each decision. Applications of heuristic techniques can potentially complement adaptive decision aiding by providing an automatic process for configuring alternatives (Weisbrod, Davis, Freedy, 1975).

2. STATE SPACE APPROACH TO PROBLEM SOLVING

The overall objective of a problem-solving system is to provide a plan of action which will accomplish some task. A task is expressed in terms of "states" which are complete descriptions of situations as they exist at some particular instant of time (Nilsson, 1971). An "action" is a transformation which, when applicable, converts one state into another. Thus, a sequence of actions ("plan" or "allocation") converts some initial state into a final, or goal, state. The expression of a problem asks the following question, "What sequence of actions can transform a given initial state into a given goal state?" In other words, "How do I get from where I am to where I want to go?" Before a problem-solving system can perform properly, it must know what actions are available, under what circumstances they can be applied, what their effects are, and what possible states can arise from their use.

In the Ballistic Missile Defense (BMD) context, a state is a combination of the current system configuration, the current environment status, and an assessment of the performance of all system components. An action is any operation that can change the state. For example, firing a missile will change the system configuration as well as the environment. Installation of new equipment may change the performance of others, etc. Once the components and conditions of a state are identified, so that complete state descriptions are possible, and once available actions have been determined, the problem-solving system can structure sequences of actions to solve any problem that might arise.

3. USING HEURISTIC SEARCH

Once a state space system description has been defined, problems can be solved using the heuristic search technique. A problem is formulated by identifying the initial state (current situation) and the goal state (desired situation). The task of the Problem Solver is to construct a sequence of actions that will transform the former into the latter.

Starting with the initial state, each possible action that could be taken is simulated and the results evaluated; that is, the new state is generated. From each of these newly generated states, possible further actions are considered respectively. This process continues until the goal state is discovered. The particular sequence of actions that led to the generation of the goal state is the solution to the problem. Thus, the Problem Solver "looks ahead" into possible future situations and evaluates future consequences arising from current actions.

The state space can be represented pictorially by imagining each state as a node and each action as an arc connecting two nodes. The development of states as taken by the Problem Solver forms a "tree"* (see Figure 2). The root of the tree is the initial state and the tips of the tree are the states under current analysis. A problem solution (sequence of actions) is a path extending outward from the root.

Generally, there will be many more nodes in the complete search tree than can fit into computer memory or can be analyzed in a reasonable

*If it is possible to arrive at a particular state by more than one path, the tree is more properly called a "graph".

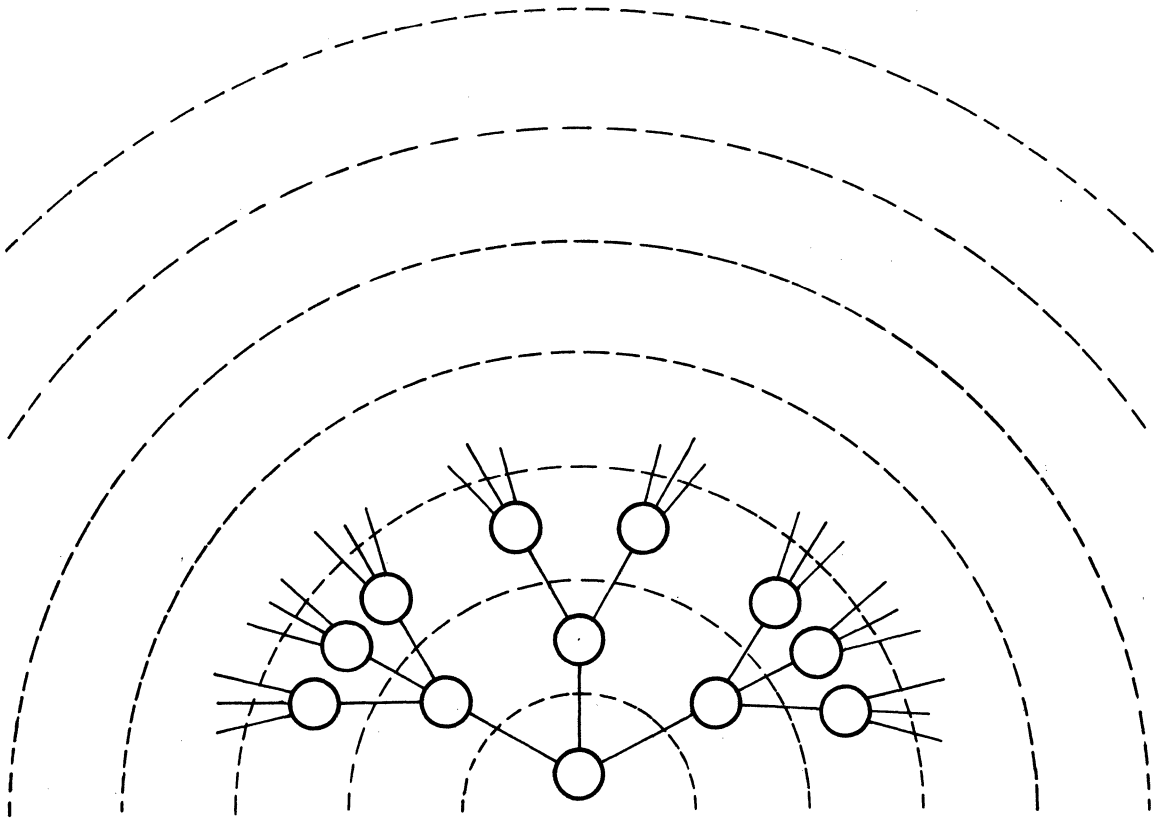


FIGURE 2. A SEARCH TREE

amount of time. For example, it has been estimated that the complete tree for the game of chess has 10^{120} nodes. One way of coping with this problem is to abstract the system under consideration so that fewer states are needed to fully describe it. This can be a great advantage, for it makes problem solving and planning possible at a much higher conceptual level. Details can be ignored which would otherwise be necessary for a purely analytical approach. But even after system abstraction, the tree is usually too large. Consequently, it is expanded as far as possible within the limits of available computer memory and the first step toward the goal (i.e., the first arc from the root) is chosen on the basis of the partially expanded tree. In order to accomplish this selection, a measure is needed that estimates the worth of each of the generated tip nodes. The selected arc at the root is the one that leads to the tip with the highest value. Such a measure is called a "heuristic evaluation function" and can use any information or data available at the particular state. In many cases, the evaluation function will be a measure of the estimated "distance" to the goal in which case the tip with the smallest value is selected. If there is just one goal state which can be reached in only one way, a bad heuristic function will delay the discovery of the solution. However, in many real-life situations, it is not the goal that is desired, but the path. That is, the "goal" consists simply of various easily reached situations at some future time. The problem is, how to get there in the best possible way.

The types of search trees described thus far have dealt only with actions that could be taken at will. In many cases, actions are taken by adversaries who are trying to do as much damage as possible, that is, trying to prevent the goal from being reached. Assuming that an adversary has access to the search tree (i.e., that he is as intelligent as we are), the best action for him to take (at each state node that permits him to take actions) is the one which leads to the minimum value of the successor nodes. Likewise, the best choices for the defense to take are those that

lead to maximum values. This is because the heuristic function is, after all, a measure of the worth of the state and the action that leads to a more valuable position is preferable. Once the heuristic function has been applied to all tip nodes, the values of the internal nodes can be calculated by taking appropriate maximums and minimums all the way back to the root node. The arc from the root that leads to the successor with the highest value is chosen as the first action in the solution. Thus, the search tree is re-generated for each step from the initial state to one of the goal states.

It is not necessary to search all possible paths in order to select the best actions. If a path to a promising tip node is discovered early, related poorer paths can be ignored. The tree thus becomes smaller and it is possible to explore further with existing computer storage. This technique for reducing the size of the tree is called "alpha-beta pruning" and allows exploration of the tree in approximately twice the depth (Nilsson, 1971).

4. INTERCEPTOR PLANNING

Heuristic search techniques can be applied to the problem of interceptor planning in a straight-forward manner. The problem is concerned with how to allocate defending interceptors to approaching enemy re-entry vehicles (RV's) as they are attacking friendly targets (such as ICBM silos, etc.). A simplified state space description would consist of the current values for each of the following quantities:

(1) The number of target silos.
(2) The number of defending interceptors.
(3) The interceptor coverage.

The coverage of an interceptor is a list of the silos that it can defend. The three possible actions are:

(1) Attack with an RV.
(2) Defend with an interceptor.
(3) Do nothing (no defense).

Action (1) is taken by the attacker (adversary) while (2) and (3) are taken by the defender and could alter the current state by either decreasing the number of available interceptors or recording a destroyed target silo respectively.* The objective of the Problem Solver is to find a defense strategy that saves some given fraction of silos.

*In this model, it is assumed that the probability of intercept as well as the probability of kill on a silo is 1.

Figure 3 shows the method of tree construction. The root node is the current state. The first layer (and each successive odd layer) corresponds to the possible attacks of the RV's: one branch for each of the silos. The second layer (and each successive even layer) corresponds to the possible defense options: one branch for each covering interceptor plus an extra one for "no intercept". It should be clear that if this tree were expanded completely, every possible offensive strategy and every possible defensive allocation would be included.

Techniques for reducing the size of the tree can be performed before problem solving is started. First, it is not necessary to plan interceptions against any RV attacking an already dead target. Second, in many cases the order of interception is not important. This means that repetitive states will emerge and can be collapsed into the same node. Third, it is assumed that the sensor system will provide information as to the targets attacked by more than one RV at a time. The number "seen in the sky" will vary with respect to the RV spacing. This means that the first few layers of the tree will have only one branch extending from nodes in the attack layers. Thus, the sensor information serves to divide the tree into two parts. Intercept planning in the first part can be done with information about the current attack. The second part of the tree is the "look-ahead" portion which poses possible future situations and evaluates consequences based on simulated actions. There is no loss in generality by assuming that the RV's are arriving in sequence. If two or more arrive simultaneously, an arbitrary sequence can be assigned to them without affecting the problem-solving procedure.

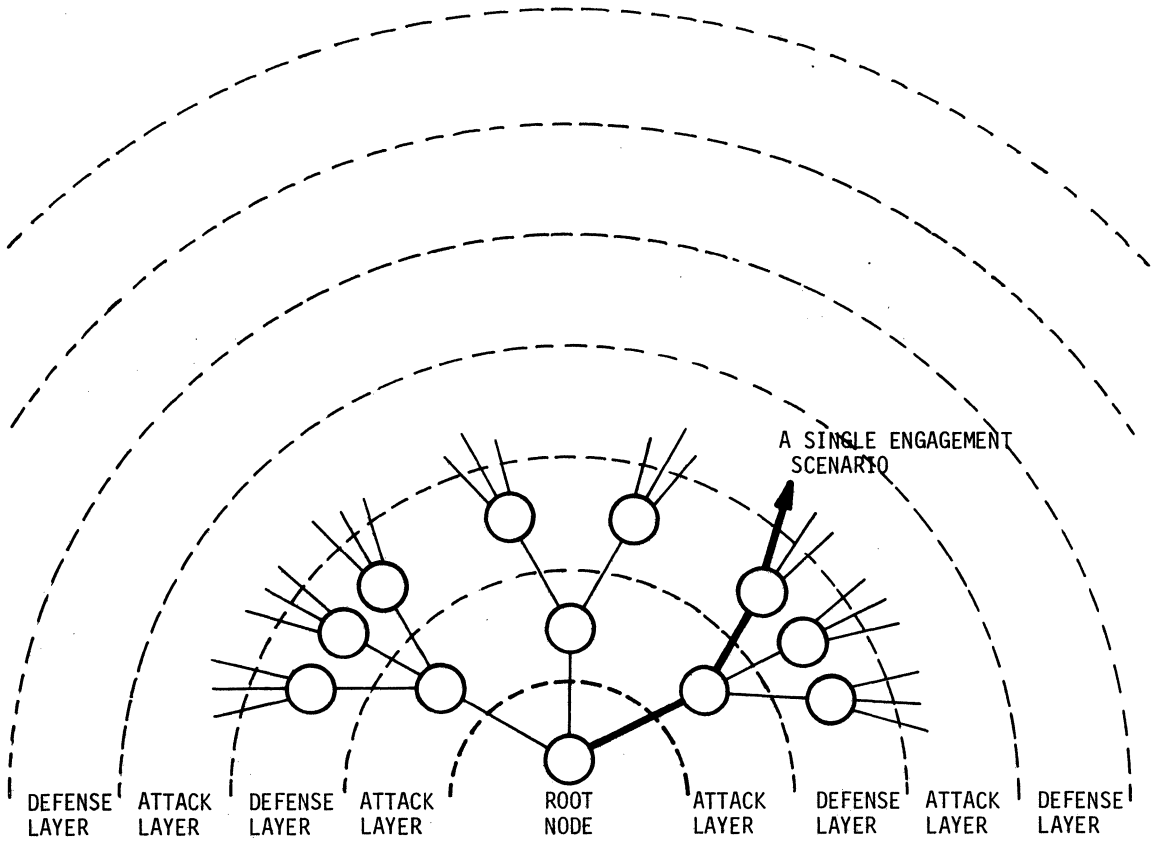


FIGURE 3. INTERCEPTOR PLANNING TREE

PERCEPTRONICS

5. THE HEURISTIC EVALUATION FUNCTION

A major component of the problem solving system is the heuristic function. In the example described above, the heuristic function must produce a value which indicates the worth of the current state. That is, it must tell the Problem Solver how well it is doing. The absolute magnitude of the value is of no importance. Only the relative value is of significance since only maximum and minimum values are compared.

For example, the following parameters would be useful in the heuristic function used for a model of interceptor planning.

S	The number of remaining silos
R	The estimated number of remaining RV's
I	The number of remaining interceptors
C	The total interceptor coverage. That is, the sum of the number of silos that each interceptor can cover (including redundancies).
S_T	Total number of initial silos
R_T	Total estimated number of RV's at start
K	A ratio between 0 and 1 indicating the desired proportion of silos saved.

The following are some candidate heuristic functions:

(1) $h = S$ — A very simple measure which says, in effect, "Save ICBM silos regardless of other considerations".

(2) $h = 2S+C$ — Places more emphasis on saving silos than on interceptor coverage.

(3) $h = \frac{ISC}{R}$ — Treats silos saved, interceptors saved, remaining coverage, and expected remaining RV's equally.

(4) $h = \frac{R_T(S-kS_T)}{S_T(1-k)} - R$ — Given the ratio k ($0 \leq k < 1$) of expected number of silos saved, this heuristic function allows silos to be selectively destroyed so that kS_T are surviving after the battle.

Heuristic function (1) would probably cause all of the interceptors to be launched early since it favors saving silos. In some situations, this may be a poor heuristic since many silos may be undefended near the end of the battle which is beyond the scope of the "look-ahead" portion of the search tree. Functions (3) and (4), on the other hand, consider factors such as the interceptor coverage and the estimated size of the attack.

6. A DETAILED EXAMPLE

The following very simple example will follow the heuristic search procedure step-by-step to show exactly how decisions are made in a BMD environment. Figure 4 depicts a hypothetical engagement in which 2 silos (S1 and S2) defended by 2 interceptors (I1 and I2) are being attacked by 3 incoming RV's. The lines connecting the interceptors to the silos indicate the coverage. It is assumed, for example, that interceptor I2 is too far away from silo S1 to protect it. Thus the coverage parameter C is equal to 3 in the current situation (i.e., initial state). The attack pattern of the RV's is predetermined and is S1, S2, S1, but for this example the radar will only be able to see one at a time.

The first branches from the root in the search tree lead to the possible targets which could be attacked. In this case, the radar can see that the first RV is heading toward S1. Therefore, only one branch is necessary (see Figure 5). Continuing with this information, there are 2 alternatives from which to choose:

(1) Launch interceptor I1
(2) Do nothing

Thus, two branches should extend outward from the S1 node to the next level (see Figure 6). "No defense" is indicated by an "x". At this point, since the radar cannot see the next incoming RV, the "look-ahead" portion of the tree is started. One node is generated for each possible attack situation. Since there are two possible targets, <u>each</u> of the previous nodes has 2 branches extending into the next level (see Figure 7).

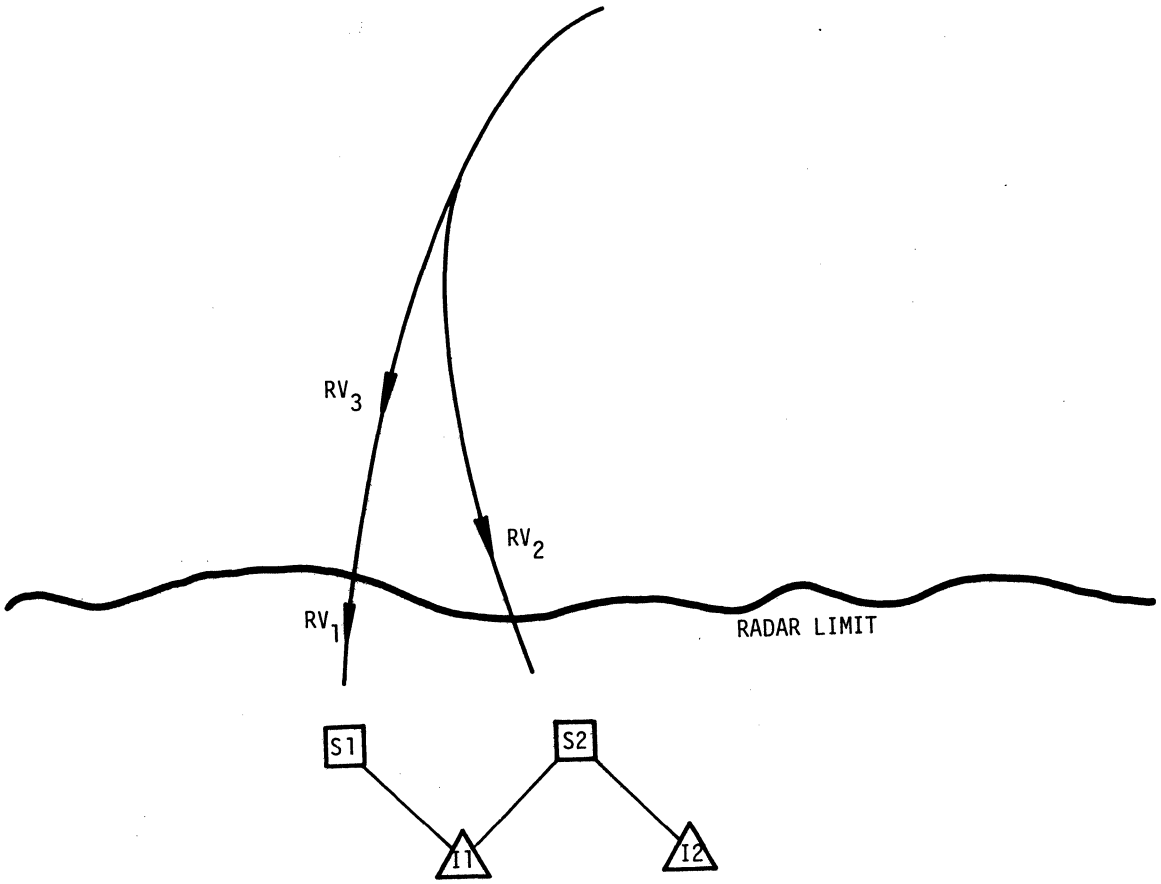


FIGURE 4. HYPOTHETICAL ENGAGEMENT

PERCEPTRONICS

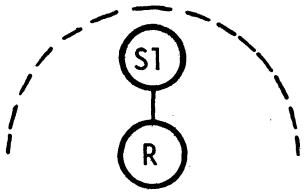


FIGURE 5. INITIAL SEARCH TREE

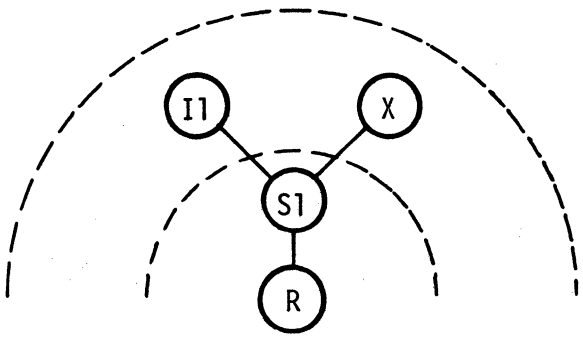


FIGURE 6. INTERCEPTOR OPTIONS

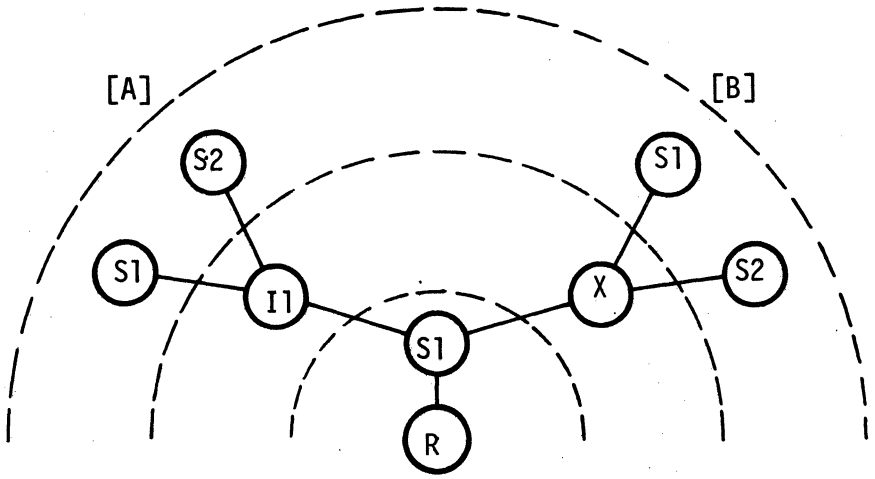


FIGURE 7. ATTACK OPTIONS

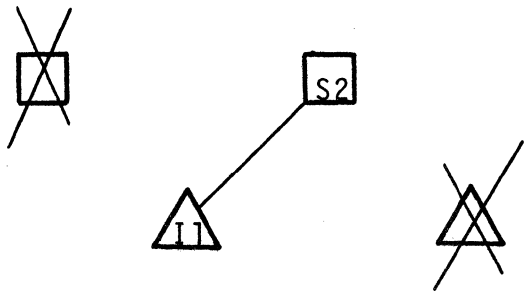
PERCEPTRONICS

The fourth level of the tree, like the second, enumerates the possible defenses against each of the postulated situations. For example, the path leading to [A] in Figure 7 supposes that S1 was attacked and defended by I1, followed by an attack on S2. Since I1 is gone, only I2 remains. Thus, 2 branches would extend into the next level: I2 and "no defense". At [B], S1 was attacked, sacrificed, and attacked again. On this path, no further defense is needed, thus, only the "x" branch would extend into the next level. Figure 8 shows the complete tree for 3 RV's.

Once the tree has been generated, the heuristic function is applied to the tip nodes. The function that is used in this example is $h = 2S+C$ where S is the number of remaining silos and C is the remaining coverage. For the path marked [A] in Figure 8, the supposed situation is:

(1) Silo S1 attacked and sacrificed
(2) S1 attacked, again, no defense needed
(3) S2 attacked and I2 defends.

The resulting situation is:



thus, S=1 and C=1 yielding a value of 3 for the heuristic function h.

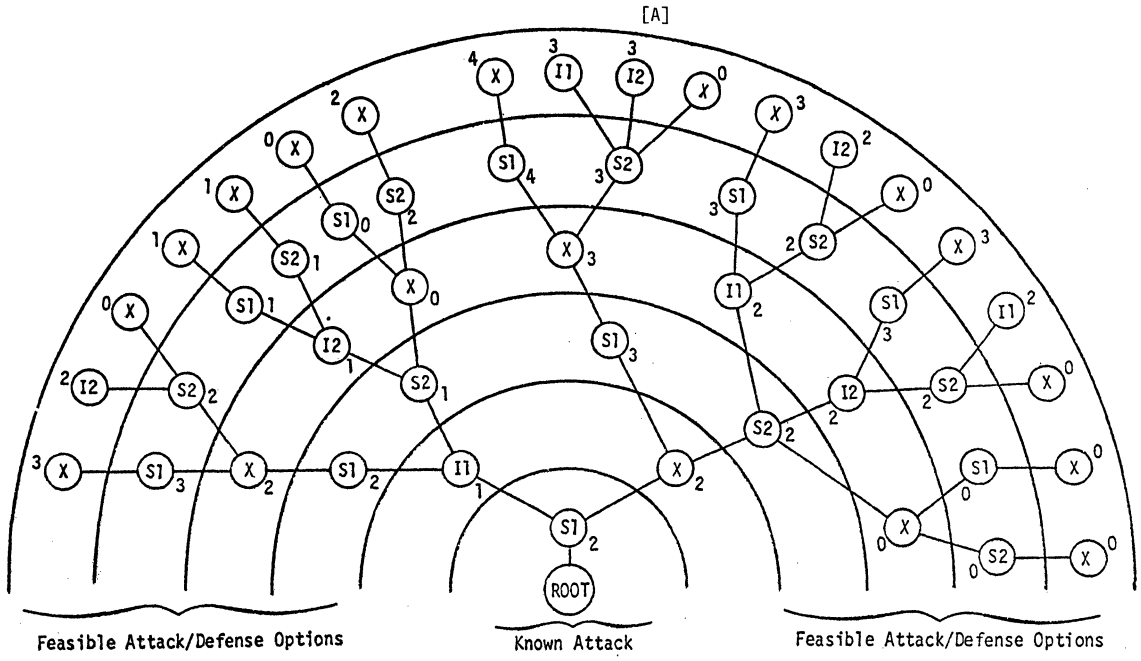


FIGURE 8. COMPLETE SEARCH TREE

The "minimum/maximum" calculations are now performed. The value of each "S" node is the maximum of the values of its successors. This is because a choice of defenses is possible. Similarly, the value of each "I" node is the minimum of the values of its successors. Here, the opponent wishes to do as much damage as possible. When the calculations are made back to the root, the branch with the highest value is chosen as the decision. In the example (Figure 8), the "no defense" branch has a value of 2 while the I1 branch has only 1. Thus, the decision is made to sacrifice silo S1. This seems reasonable since S2 still has a high coverage level.

With the first decision made, the Problem Solver waits until the radar provides information on the target of the next RV. When this is known, a new tree is generated. Figure 9 shows this tree assuming that S2 is now the target. Thus, a new tree is constructed for each single decision that is made. The tree does not, of course, have to be expanded completely in order to apply the heuristic function. Any situation complex enough to be useful will have a tree too big either to fit into computer memory or to be processed in a reasonable amount of time. Consequently, it is expanded as far as available time and memory will allow. As can be seen in Figure 9, the Problem Solver is indifferent between defending with I1 or I2 but does not sacrifice S2. With the last RV heading for S1, no defense is needed.

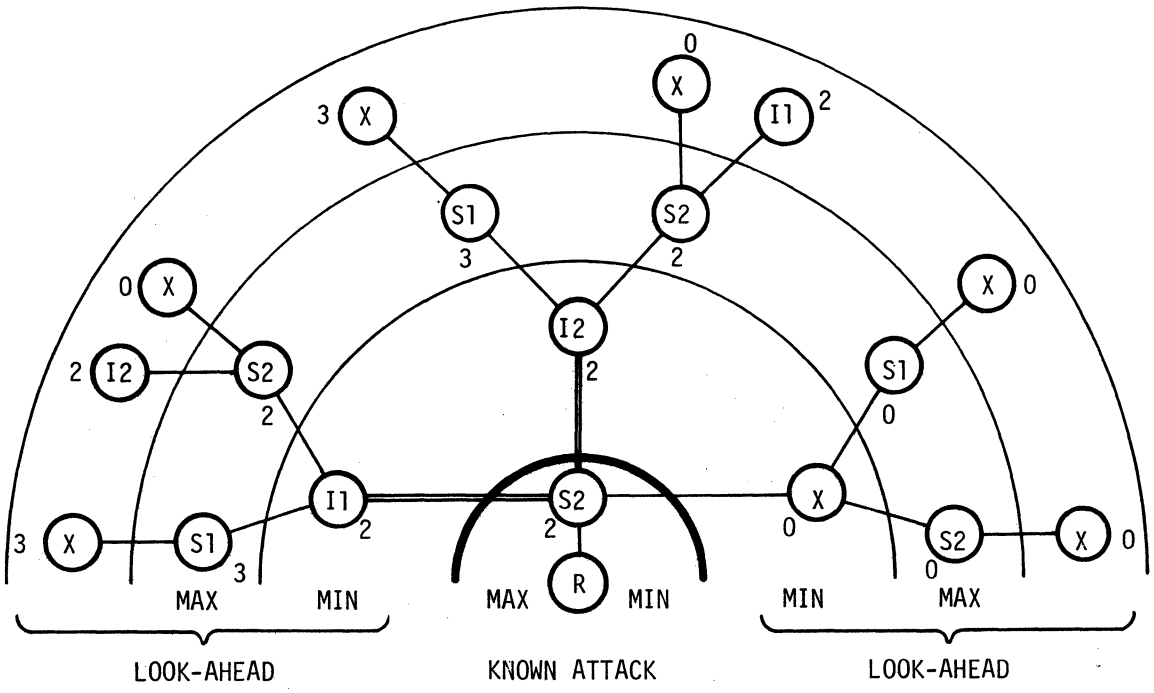


FIGURE 9. SECOND DECISION

PERCEPTRONICS

7. THE ALPHA-BETA ALGORITHM

The alpha-beta pruning technique is a method of tree expansion that allows large portions of the tree to be ignored if it can be proven that a complete search would not alter the ultimate decision. The technique relies on two important search properties: (1) successive maximum and minimum calculation on alternate layers of the tree, and (2) a depth-first expansion algorithm.

The expansion algorithm described previously is "breadth-first" in the sense that all possible nodes extending from the root are generated before any second-level nodes are generated (see Figure 10). A "depth-first" expansion generates nodes along a single path all the way to the end of the tree (or to some predefined depth bound). Then, brother nodes are generated until no more exist (see Figure 11). The advantage of depth-first expansion is that the heuristic function can be applied as the tree is being generated. When a value is returned, it can be immediately assigned to successive predecessor nodes back to the root. As further expansion occurs, these preliminary maximums and minimums indicate the best solutions as they are found.

The alpha-beta technique takes advantage of these preliminary values. Consider a single node in the middle of the tree (Figure 12, node A) that receives its value from the maximum of its successors. As its successors are being expanded, their values are obtained from the minimum of their successors (node B). Therefore, the values of the immediate successors can only get lower. If it should happen, that the value of one of the successors gets lower than the current preliminary value of the main node, there is no use following any more paths since a formerly expanded tree portion would be the prime candidate for selection.

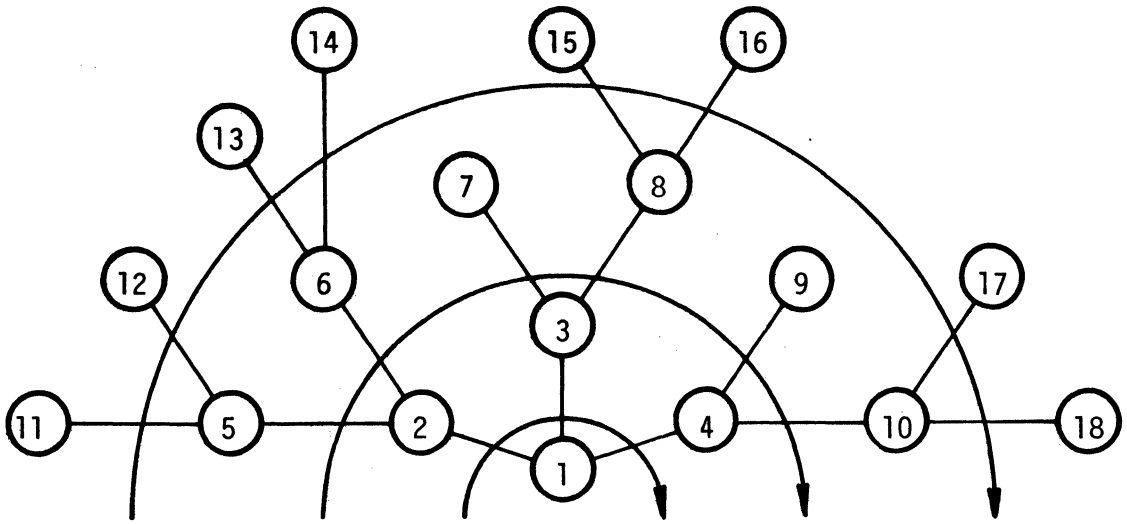


FIGURE 10. BREADTH-FIRST EXPANSION ORDER

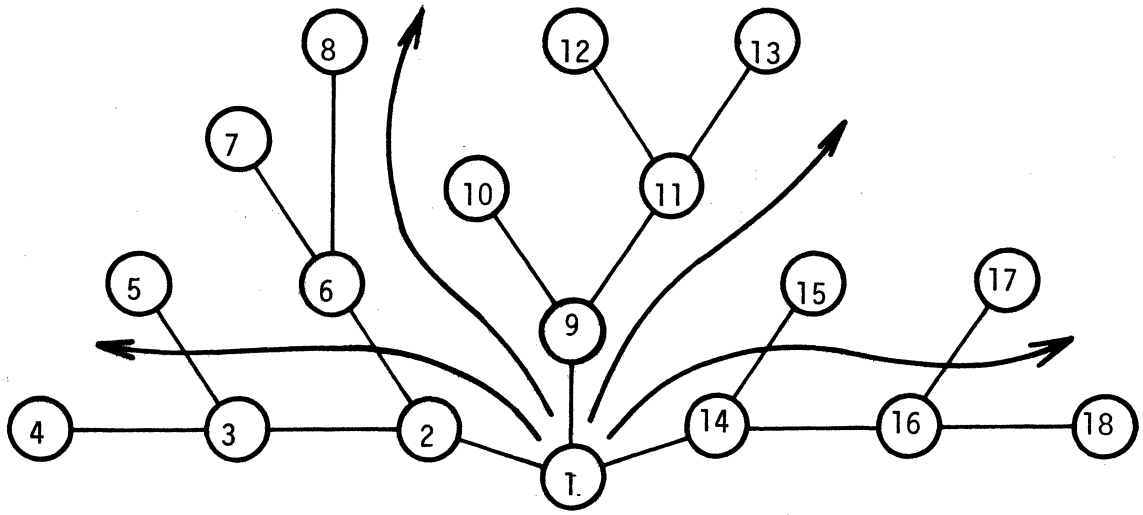


FIGURE 11. DEPTH-FIRST EXPANSION ORDER

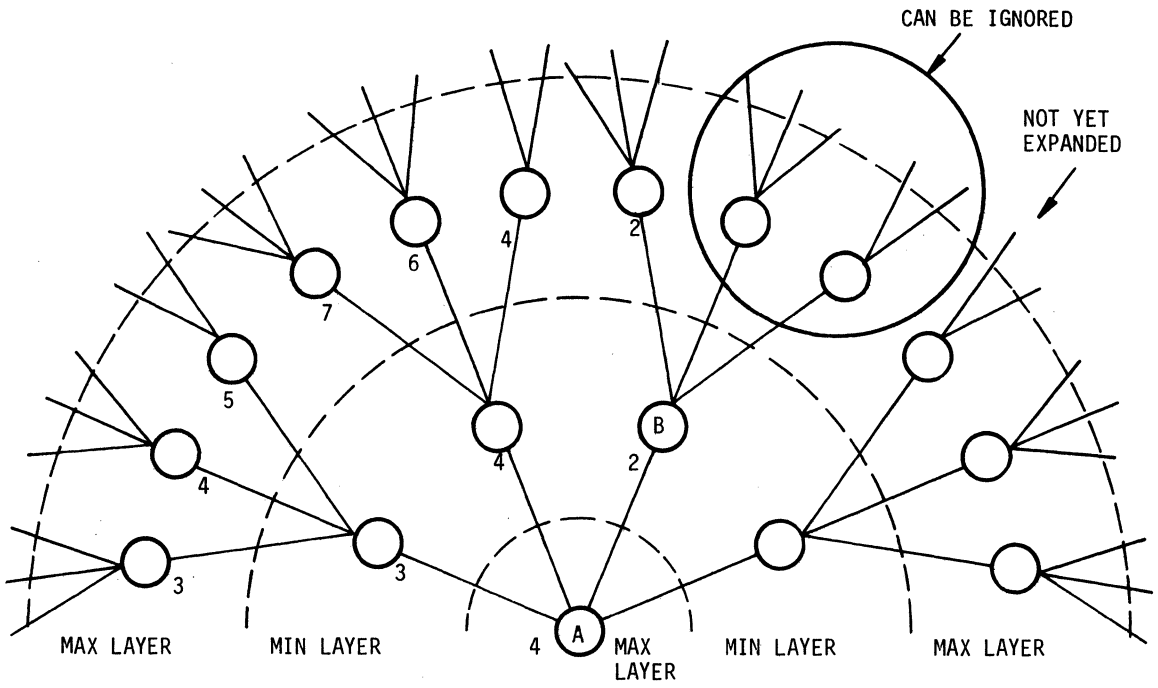


FIGURE 12. ALPHA-BETA CUTOFF

PERCEPTRONICS

Figure 12 shows an example of this. After expanding 2 successor branches, node A has a preliminary maximum of 4. During the expansion of the third branch, a value of 2 is obtained at node B. Since node B takes minimums, it can only get lower than 2. But node A already has a value of 4 and would not choose node B as the best decision. Therefore, all other branches leading out from node B can be ignored.

This pruning is the "alpha" half of the alpha-beta. The "beta" half operates in precisely the reverse for nodes in the minimum layers. By using the alpha-beta algorithm, the tree can be explored approximately twice as deep in the same amount of time.

8. A SIMULATOR DESIGN

The following section describes a detailed design for a program that simulates a missile engagement and uses heuristic search to allocate defenses. It is independent of data structure and assumes that the necessary data is available and can be updated easily. Figure 13 shows the overall components and their relationships. The problem solver is actually a pair of mutually recursive functions that expands the tree implicitly through subroutine calls. When a preset depth bound is reached, the heuristic function evaluates the current state.

The following variables are used in the flow chart description.

- V — A local variable used to store changing ALPHA and BETA values.
- A — Stores the result returned from the ATTACK routine.
- B — Stores the result returned from the DEFEND routine.
- R — The current number of RV's left.
- R_T — The total number of RV's.
- D — The current tree depth level.
- S — A list of live silos plus the current silo under attack.
- $\hat{S}$ — An updated version of S with possibly a destroyed silo.
- I — A list of interceptors currently available with their coverage.
- $\hat{I}$ — An updated version of I with possibly a launched interceptor.
- I_Q — The chosen interceptor to be launched.
- h — Stores the result from the heuristic function.

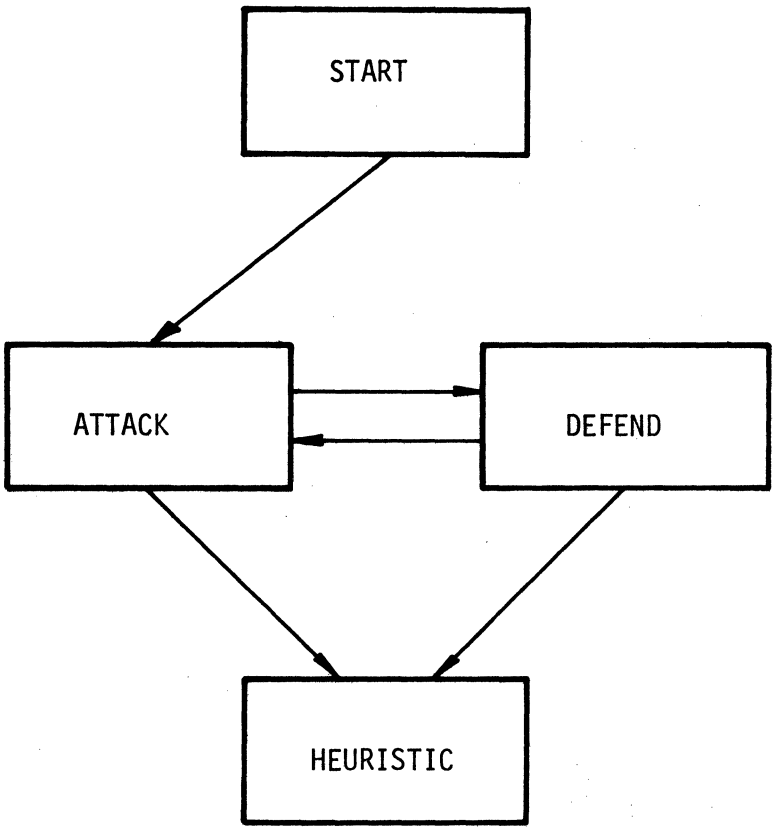


FIGURE 13. SIMULATOR COMPONENTS

The START routine (Figure 14) is the supervisor for the simulator and keeps a record of the best action as the tree is being generated. It begins the first tree layer by iterating through the interceptors covering the first target. With each assumed intercept, it calls the ATTACK routine for the second RV. During the iteration, the interceptor that produces the highest heuristic value is noted and when the analysis is complete, it is launched. Of course, it may happen that "no defense" is best. The entire process is repeated for each new RV.

The ATTACK routine (Figure 15) receives the following parameters as input: (1) the current depth, (2) the current silo state, (3) the current interceptor resources, (4) the expected number of RV's left, and (5) a BETA value which will determine if search is to be halted. If the RV's have been exhausted (by simulation) or if the depth bound has been reached, the heuristic function is called and its value is the value of ATTACK. Otherwise, the DEFEND subroutine is called. If the target is known, DEFEND need only be called once. If the target is unknown, DEFEND must be called for each of the possible target silos (i.e., the look-ahead portion). During the DEFEND iteration, the results are continually being compared with former results in order to find the minimum value. If this descending value drops below the Beta input value, search is suspended.

The DEFEND subroutine (Figure 16) operates in almost an identical manner. The only difference is that search is suspended when the increasing heuristic value exceeds the Alpha cutoff parameter.

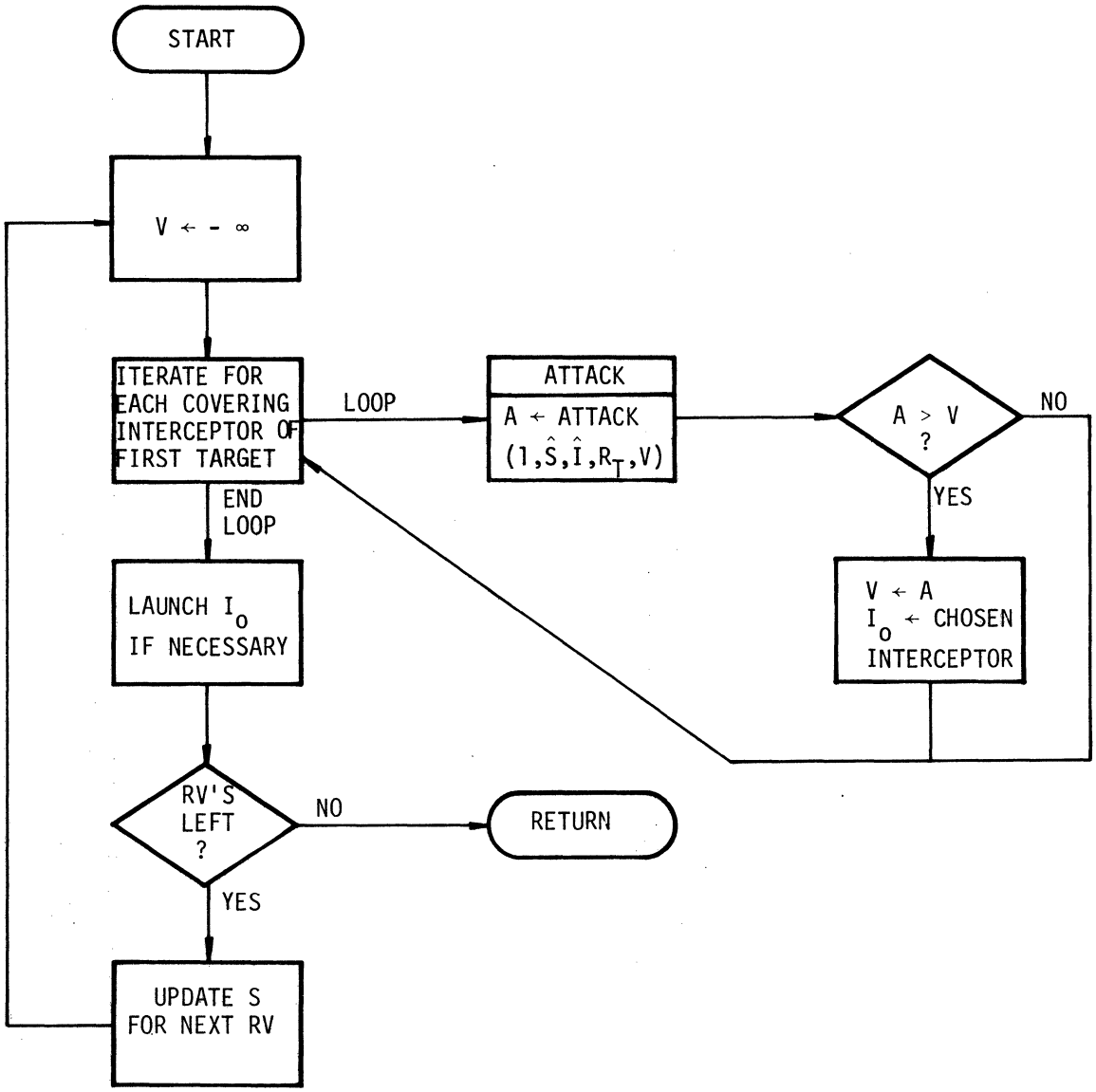


FIGURE 14. START ROUTINE

PERCEPTRONICS

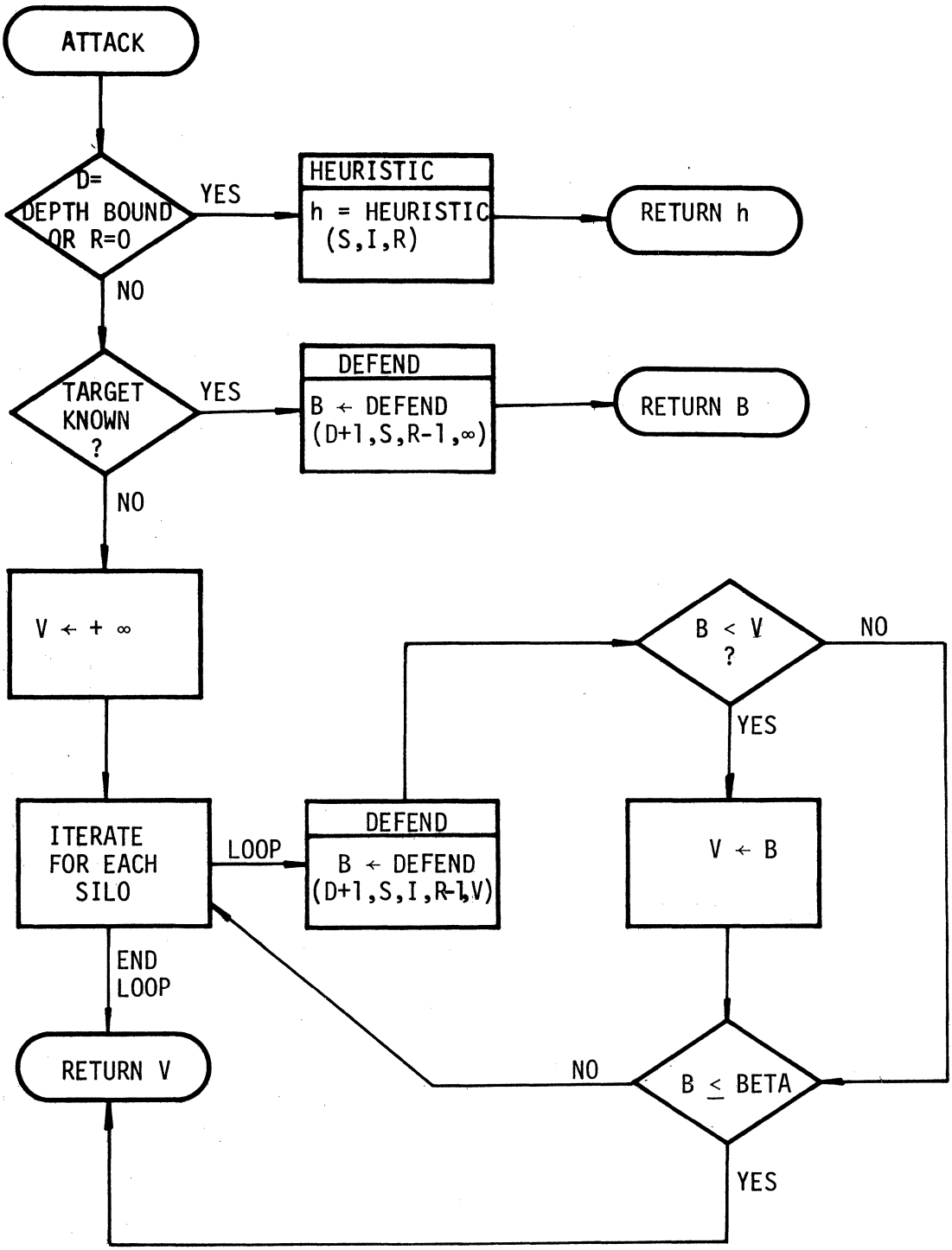


FIGURE 15. ATTACK (D, S, I, R, BETA)

PERCEPTRONICS

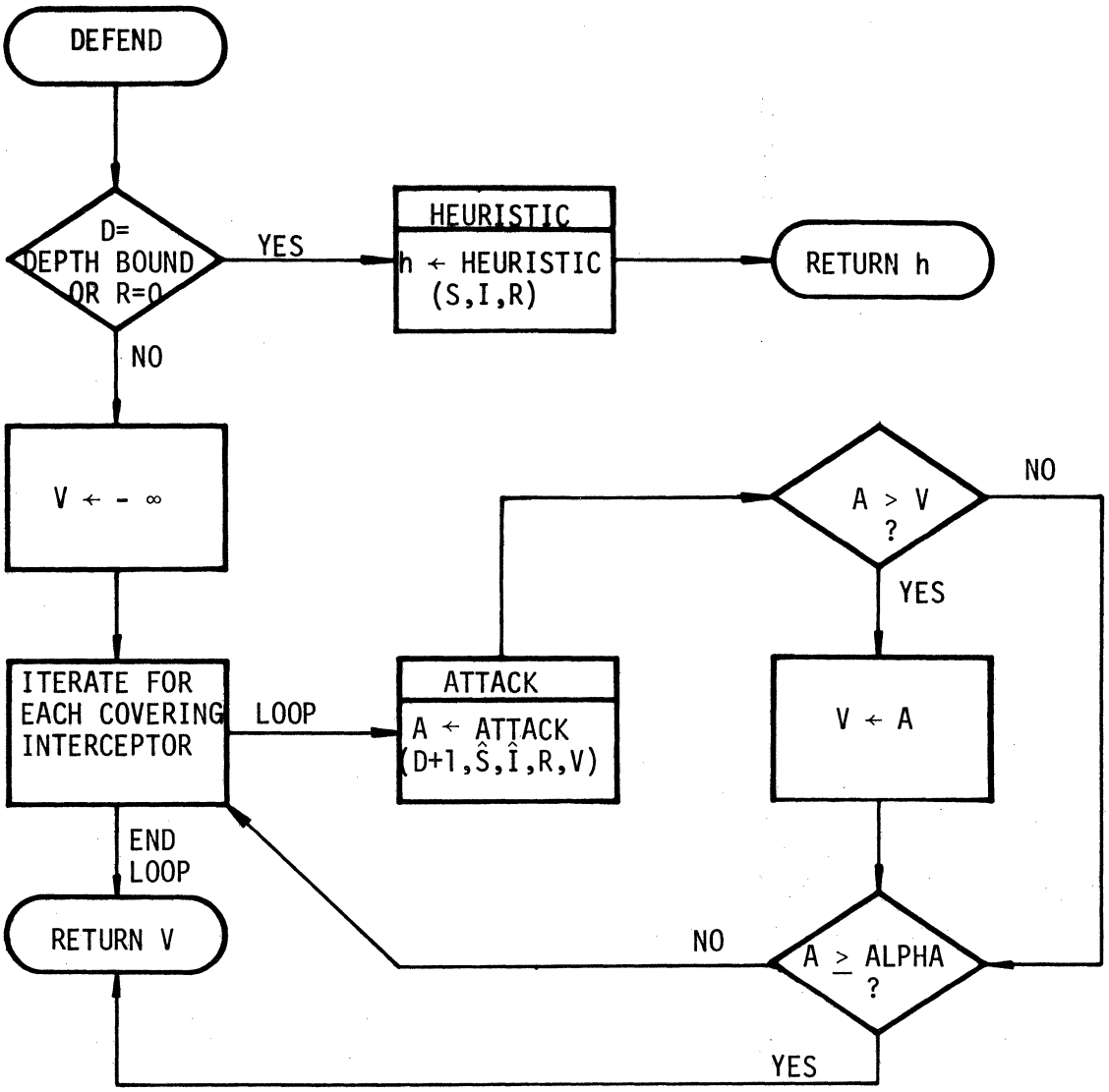


FIGURE 16. DEFEND (D, S, I, R, ALPHA)

PERCEPTRONICS

9. SIMULATION RESULTS

The results of a simulated engagement involving 7 silos defended by 12 interceptors against 18 attacking RV's is summarized in Tables 1-6. The performance of each of 5 candidate heuristic functions was measured with respect to 3 varying levels of coverage. The 5 heuristic functions are:

$$H_1 = SC^2$$

$$H_2 = SI^2$$

$$H_3 = \frac{ICS}{R}$$

$$H_4 = SC$$

$$H_5 = S$$

The attack pattern, chosen initially at random, remained the same for each of the simulation runs. Numbering each of the silos from 1 to 7, the 18 RV-attack pattern is as follows:

6, 6, 7, 2, 3, 7, 6, 5, 1, 1, 4, 4, 5, 5, 2, 2, 1, 7

The alpha-beta search technique described in the previous section was used with a look-ahead depth of 5 and information about the target of only one RV at a time.

The first series of simulation runs used a light coverage relation between interceptors and silos. Table 1 shows the coverage in matrix form. Table 2 presents the step-by-step interceptor defense strategies selected by each one of the heuristic functions. A dash indicates no defense. Tables 3, 4, 5, and 6 present similar results for medium and heavy coverage.

COVERING INTERCEPTOR NUMBER	SILO NUMBER 1	2	3	4	5	6	7
1	*						
2	*	*					
3	*	*					
4		*					
5		*	*				
6			*				
7			*	*			
8				*			
9				*	*		
10					*		
11					*	*	
12						*	*

TABLE 1. LIGHT COVERAGE

RV#	ATTACKED SILO #	H_1	H_2	H_3	H_4	H_5
1	6	--	--	--	--	12
2	6	--	--	--	--	11
3	7	12	12	12	12	--
4	2	5	5	5	5	5
5	3	6	7	6	6	7
6	7	--	--	--	--	--
7	6	--	--	--	--	--
8	5	11	11	11	11	10
9	1	1	--	1	1	3
10	1	3	--	3	3	2
11	4	9	--	--	9	9
12	4	8	--	--	8	8
13	5	10	10	10	10	--
14	5	--	9	9	--	--
15	2	4	4	4	4	4
16	2	2	3	2	2	--
17	1	--	--	--	--	1
18	7	--	--	--	--	--
SILOS SAVED:		3	3	3	3	3
INTERCEPTORS SAVED:		1	4	2	1	1

TABLE 2. DEFENSE STRATEGIES; LIGHT COVERAGE

COVERING INTERCEPTOR NUMBER	SILO NUMBER 1	2	3	4	5	6	7
1	*						
2	*	*	*				
3	*	*	*				
4		*	*	*			
5		*	*	*			
6			*	*	*		
7			*	*	*		
8				*	*	*	
9				*	*	*	
10					*	*	*
11					*	*	*
12							*

TABLE 3. MEDIUM COVERAGE

RV#	ATTACKED SILO	H_1	H_2	H_3	H_4	H_5
1	6	11	--	--	11	11
2	6	10	--	--	10	10
3	7	12	12	12	12	12
4	2	5	5	5	5	5
5	3	7	7	7	7	7
6	7	--	11	--	--	--
7	6	6	--	--	9	9
8	5	9	10	11	8	8
9	1	3	--	--	3	3
10	1	1	--	--	2	2
11	4	--	--	--	--	6
12	4	--	--	--	--	4
13	5	8	9	10	--	--
14	5	0	8	9	--	--
15	2	--	--	--	--	--
16	2	--	--	--	--	--
17	1	--	--	--	1	1
18	7	--	--	--	--	--
SILOS SAVED:		2	2	2	3	4
INTERCEPTORS SAVED:		2	5	6	2	0

TABLE 4. DEFENSE STRATEGIES; MEDIUM COVERAGE

COVERING INTERCEPTOR NUMBER	SILO NUMBER 1	2	3	4	5	6	7
1	*	*	*	*			
2	*	*	*	*			
3	*	*	*	*			
4		*	*	*	*		
5		*	*	*	*		
6		*	*	*	*		
7			*	*	*	*	
8			*	*	*	*	
9			*	*	*	*	
10				*	*	*	*
11				*	*	*	*
12				*	*	*	*

TABLE 5. HEAVY COVERAGE

RV#	ATTACKED SILO	H_1	H_2	H_3	H_4	H_5
1	6	12	--	--	12	12
2	6	11	--	--	11	11
3	7	--	12	12	10	10
4	2	6	6	6	6	6
5	3	9	9	9	9	9
6	7	--	--	--	--	--
7	6	--	--	--	8	8
8	5	10	11	11	7	7
9	1	--	--	--	--	3
10	1	--	--	--	--	2
11	4	8	--	10	5	5
12	4	7	--	8	4	4
13	5	--	10	--	--	--
14	5	--	8	--	--	--
15	2	5	--	--	--	--
16	2	4	--	--	--	--
17	1	--	--	--	--	1
18	7	--	--	--	--	--
SILOS SAVED:		3	2	2	3	4
INTERCEPTORS SAVED:		3	6	6	3	0

TABLE 6. DEFENSE STRATEGIES; HEAVY COVERAGE

10. CONCLUDING REMARKS

The remarkable improvement in defense levels gained by using heuristic search is mainly due to the adaptive nature of the algorithm. Defense strategy allocations are planned and executed while the battle is in progress. Information gained through engagement simulations can be used directly to improve the performance of heuristic functions.

It is interesting to consider the position taken by the algorithm as it plans a defense. The look-ahead portion of the algorithm assumes that the enemy approaching RV's have the intelligence and the option of choosing the target which will disrupt the defenses most at re-entry time. This, of course, is clearly unrealistic since attack strategies are pre-planned. However, this only means that the overall performance will be better than that shown thus far. It may be said that the heuristic search process plans for a battle that occurs over a very long period of time. A period which is so long that the enemy has a chance to choose the target of each RV after the results of the previous one are known. On the other hand, among the group of postulated attacks are those known as "saturation attacks" in which great numbers of RV's approach at once. In such cases, again, the adaptive process can do better than expected because it has a great deal of information as to the actual targets attacked. Planning can be done on this basis to a much more accurate degree and with a smaller search tree.

Adaptive logical processes such as heuristic search have wide application in solving BMD problems. They can adapt to a changing environment, adapt to internal changes, learn from experience, and take advice from experts. They can be used effectively to solve problems in areas such as command and control, resource and manpower allocation, radar

reconfiguration, computer communication network planning, fault detection, etc. Decisions in these and other important BMD areas, formerly made by human intuition, can be analyzed and executed with greater reliability and confidence.

11. REFERENCES

1. Fikes, R.E., Hart, P.E., and Nilsson, N.J. Learning and Executing Generalized Robot Plans. Artificial Intelligence 3, North-Holland Publishing Company, 1972, pp. 251-288.

2. Nilsson, N.J. Problem-Solving Methods in Artificial Intelligence. McGraw-Hill, 1971.

3. Slagle, J.R. Artificial Intelligence: The Heuristic Programming Approach, McGraw-Hill, 1971.

4. Simon, H.A. and Kadane, J.B. Optimal Problem-Solving Search: All or None Solutions. Artificial Intelligence, North-Holland Publishing Company, Fall 1975, 6(3):235-248.

5. Weisbrod, R., Davis, K., and Freedy, A. Adaptive Utility Assessment in Dynamic Decision Processes: An Experimental Evaluation of Decision Aiding. Proceedings of the 1975 International Conference on Cybernetics and Society. IEEE, 1975, pp. 302-309.