Naval Research Laboratory

Washington, DC 20375-5000

(2)

NRL Memorandum Report 6289

Algorithm Suite for Defensive Weapons Allocation

S. BRAVY

Integrated Warfare Technology Branch
Information Technology Division

W. A. METLER AND F. L. PRESTON

AT&T Bell Laboratories
Whippany, New Jersey 07981

November 8, 1988

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) NRL Memorandum Report 6289	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Research Laboratory 6b. OFFICE SYMBOL (If applicable) Code 5579	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code) Washington, DC 20375-5000	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code)	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 63223G; PROJECT NO.; TASK NO.; WORK UNIT ACCESSION NO. DN155-097

11. TITLE (Include Security Classification)
Algorithm Suite for Defensive Weapons Allocation

12. PERSONAL AUTHOR(S)
Bravy, S., Metler, W.A. and Preston, F.L.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Final	FROM _____ TO _____	1988 November 8	18

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Allocation, Marginal Returns, Greedy

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The engagement problem faced by a battle manager acting within a defensive system against a ballistic missile threat is described and modeled. A strategy for solving the engagement problem with a suite of weapons allocation algorithms is discussed. The individual algorithms are described; since the weapons allocation problem has been shown to be NP-Complete, all the algorithms employ heuristics.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Steve Bravy	22b. TELEPHONE (Include Area Code) (202) 767-3919 22c. OFFICE SYMBOL Code 5579

DD FORM 1473, 84 MAR 83 APR edition may be used until exhausted. All other editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE

CONTENTS

ALGORITHM SUITE FOR DEFENSIVE WEAPONS ALLOCATION

Introduction

We wish to design an algorithm or a suite of algorithms that can be used by a defensive battle manager to allocate defensive weapons to offensive threats. This problem has been modeled before. In the 1950s the Prim-Read algorithm was described [1]. It produced deployments of interceptor missiles and their associated firing plans to defend a collection of separated point targets against an attack by an unknown number of sequentially arriving ballistic missiles. The scheme involved deploying and firing the interceptors in a manner that equalized the probabilities that each of a prescribed number of attacking weapons destroys the target. More recently, Burr, et al. [2], showed that the Greedy Algorithm when applied to the Prim-Read model produces a globally optimal integral solution. The objective function for their study was to minimize the total number of interceptors required to defend K targets against an attack of N missiles, subject to a given upper bound on the maximum target value destroyed per attacking weapon, where neither N nor upper or lower bounds on N are known to the defender in advance.

There are two types of defenses, point and area. Point defenses consist of weapons dedicated to the defense of particular targets, while area defenses consist of weapons used to shoot at objects in an area (or volume) regardless of the object's intended target. The Prim-Read defense focuses on deploying a terminal defense and assumes that the firing plans of the defending interceptors will be used in a point defense mode. There is, however, the possibility of three tiers of defense -- boost phase, midcourse phase, and terminal phase -- and during the boost and midcourse phases space based weapons potentially may defend against any missiles in range, i.e. an area defense mode. The area defense mode of operation also characterizes submarine based interceptors, air defenses involving ground to air and ship to air projectiles, and theater level defenses against short range missiles carrying conventional warheads.

We model the problem of allocating defensive weapons for area defense after the threat is known, whereas the Prim-Read model addresses the problem of deploying defensive weapons for point defense before the threat is known. Our goal is to determine the interceptors' best firing schedule over time to defend the target sites against the offensive missiles. The best schedule is one that maximizes the total expected surviving values of the sites. There are certain shot limits, and the kill probabilities depend on the missile, the interceptor, and the firing time.

First we describe a general model of the allocation problem we are addressing, and a sequence of simplifying assumptions to make the problem more tractable. For each special case we point out the properties of the objective function and constraints that determine what solution strategies are feasible. The algorithms that are used to solve each case take advantage of the special circumstances of the situation. Then we focus on the particular case we have chosen to solve, and present the algorithm suite used to solve it.

Assumptions and Notation

The general problem is that of a defensive battle manager that must allocate defensive weapons to offensive threats to maximize the surviving value of targeted assets. For purposes of generality we

Manuscript approved August 29, 1988.

define a "weapon" as having an inventory of "shots" and a "threat object" to be anything potentially causing damage to defensive assets. The purpose of this section is to introduce overall assumptions and notation.

a. We assume that there are K targets that are Blue assets (population centers, command and control locations, communications nodes, ICBM launch sites, etc.), and that each target k has a value, w_k, assigned by the defense.

b. We assume that a weapon may be used against threat objects aimed at different targets, that multiple weapons may be used against the same threat object, that the destruction of one threat object is independent of the destruction of another threat object, and that the defense knows how many threat objects there are during an attack and the target of each threat object.

c. We assume the attacker's strategy is known to the defense and can be expressed by:

$$\{ J_1, \cdots, J_K \}$$

where J_k denotes the set of threat object indices targeted at k. The offensive threats may not change during an engagement. We define a *firing plan* to be a set of values α_{ijt}, where α_{ijt} is the number of shots from weapon i allocated to threat object j in time period t. The α_{ijt} are strictly nonnegative integer values. For scheduling purposes time is divided into discrete periods, t_0, t_1, ..., T. Shots are fired at specific objects at specific periods. Once fired they cannot be redirected. There are upper bounds on the number of shots that can be fired from a weapon over all time, and from a weapon in each time period.

d. Let p_{ijt} denote the probability that a shot from weapon i fired at time t kills threat object j. We assume that these probabilities may be different among the weapon/object/time period combinations, and some p_{ijt}s may be zero.

We now present a sequence of four defensive weapons allocation models, each with successively more assumptions that make the problem simpler and more tractable. The specific assumptions that underlie each model are identified.

Weapons Allocation Models

General Model

As the threat objects become identified with their targets, the value of destroying the object is related to the surviving value of the target at present and in the future. A particular assignment of weapons to objects may entail a whole set of shot-object encounters, each with a probability of successful destruction of the object. The result of all these encounters is a random event -- a set of "leaking" threat objects to damage defensive assets. The objective, then, is to find an allocation that maximizes the total expected surviving value of the targets under attack. This objective function can be expressed by the following general formulation:

$$\max S = \sum_k w_k \left\{ \prod_{j \in J_k} \left[1 - \prod_{i,t} (1 - p_{ijt})^{\alpha_{ijt}} \right] \right. \tag{1}$$

$$+ \sum_{m \in J_k} f_k(m) \prod_{\substack{j \in J_k \\ j \neq m}} \left[1 - \prod_{i,t} (1 - p_{ijt})^{\alpha_{ijt}} \right] \left[\prod_{i,t} (1 - p_{imt})^{\alpha_{imt}} \right]$$

$$+ [\ \textit{fractional residue when exactly 2 threat objects get through}\ ]$$

$$\left. + \cdots + \left[f_k(all j \in J_k) \prod_{j \in J_k} \prod_{i,t} (1 - p_{ijt})^{\alpha_{ijt}} \right] \right\}$$

subject to

$$\sum_{t=0}^{T} \sum_{i} \sum_{j=1}^{N} \alpha_{ijt} \le M : \textit{ limit on total shots fired}$$

$$\sum_{t=0}^{T} \sum_{j=1}^{N} \alpha_{ijt} \le I_i : \textit{ the inventory of weapon } i$$

$$\sum_{j=1}^{N} \alpha_{ijt} \le c_{it} \quad \textit{for all } i, t : \textit{ limit on \# shots weapon } i \textit{ in time } t$$

$$\alpha_{ijt} \in Z^{+}, \textit{ positive integers and } 0$$

where

S : *expected surviving value*
$f(\,)$: *fractional damage function* $\in[0,1]$ for *specific object indices*
j : *index of threat objects*
w_k : *the value of Blue's target k*
J_k : *set of indices of threat objects going to target k*
i : *index of weapons*
k : *index of target*
α_{ijt} : *number of shots from weapon i assigned to threat object j shot at time t*
p_{ijt} : *probability of a shot from i at time t destroying threat object j*

The decision variables are the α_{ijt}s. This formulation permits specific damage outcomes to be identified with each combination of leakers. This formulation is presented for completeness, but is too complicated for practical solution.

No Partial Damage Model

When we assume any one threat object that gets through the defense destroys essentially all its target value, then all the "partial-damage" f(.) values from the general formulation (1) are 0 and we have:

$$\max S = \sum_{k \in K} w_k \left\{ \prod_{j \in J_k} \left[1 - \prod_{t=0}^{T} \prod_{i} (1 - p_{ijt})^{\alpha_{ijt}} \right] \right\} \tag{2}$$

subject to [same as (1)].

This problem probably occurs when precise target/threat object pairings are known and may occur during the final stages of an engagement. In this case one leaker causes total destruction of the target. Note that no value is obtained by allocating anything less than one shot per object going to a target. This discontinuity -- a jump from zero value to a positive expected surviving value as the last of a target's threatening objects is shot at -- causes difficulties for solution strategies.

One Threat per Target Model

If we make the further assumption that there will be at most one threat object aimed at each target, then the product over $j \in J_k$ of the no partial damage objective is reduced to one term. Now the objective is to minimize the inner product, which is the expected damage to the targets by "leakers" through the defense, given the set of α_{ijt} assignments. (Thus we denote this value by L). Since now the js and the ks are synonymous, we retain the j index instead of k.

$$\min L = \sum_{j=1}^{N} w_j \prod_{t=0}^{T} \prod_{i} (1 - p_{ijt})^{\alpha_{ijt}} \tag{3}$$

subject to [same as (1)].

This objective is most applicable when precise targeting is not known and potential target values can be aggregated across an area target. It is also applicable when object damages are small enough to be

independent of one another.

Single Shot per Threat Model

If in (3) we restrict the shot allocations to no more than one shot per threat object during the firing regime $[0,T]$, and each weapon can only fire once during any period, then the problem assumes its simplest form:

$$\max S = \sum_{j=1}^{N} \sum_{t=0}^{T} \sum_{i} w_j \, p_{ijt} \, \alpha_{ijt} \tag{4}$$

where

$$\sum_{t=0}^{T} \sum_{i} \sum_{j=1}^{N} \alpha_{ijt} \leq M\text{: } \textit{total inventory of } M \textit{ shots} \tag{4a}$$

$$\sum_{t=0}^{T} \sum_{i} \alpha_{ijt} \leq 1\text{: } \textit{one shot for each threat object } j$$

$$\sum_{j=1}^{N} \alpha_{ijt} \leq 1\text{: } \textit{one shot by each weapon } i \textit{ in any } t$$

$$\alpha_{ijt} \in \{0,1\}$$

This objective is applicable when employing a "shoot-look-shoot" firing strategy using kill assessment feedback. The battle manager directs the firing of a set of shots, looks to determine their outcomes (kill or miss), then issues more firing orders based on these outcomes, and so on. In this formulation the time periods t might correspond to weapon refire times (thus allowing only one shot per weapon per time period) and the regime $[0,T]$ might correspond to the time for intercept and kill assessment. The use of this strategy assumes the battle manager has good shots, good kill assessment, and has enough battle space to fire shots sequentially one at a time.

The problem we choose to address is (2). Before we describe out solution strategy, and as a way of motivating our approach, we describe some results relating to the solution of problems (2), (3) and (4).

Some Related Mathematical Results

Results Applicable to Model (4)

A broad class of mathematical models in operations research lead to formulations where we must determine the values of the decision variables x_j, $j \in \{0,1,2,\ldots,n\}$ to

$$\begin{aligned} &\max \textit{ or } \min x_0 = f(x_1, x_2, \ldots, x_n) \\ &\text{subject to} \\ &g_i(x_1, \ldots, x_n) \leq b_i, \; i = 1, 2, \ldots, m \\ &\qquad x_j \geq 0, \; j = 1, 2, \ldots, n. \end{aligned}$$

The function f is the objective function, while $g_i \leq b_i$ represents the i^{th} constraint, where b_i is a constant. The constraints $x_j \geq 0$ are called the nonnegativity constraints, which restrict the variables to zero or positive values only.

If f and g_i are linear functions of the decision variables x_j then the problem may be formulated as a linear program (LP):

$$\begin{aligned} &\max x_0 = \sum_{j=1}^{n} c_j \, x_j \\ &\text{subject to} \\ &\sum_{j=1}^{n} a_{ij} \, x_j \leq b_i, \; i = 1, 2, \ldots, m \\ &\qquad x_j \geq 0, \; j = 1, 2, \ldots, n. \end{aligned}$$

Formulation (4) qualifies as a linear program. In particular, if the shot limit and inventory constraint (4a) is removed or strict equality holds, the problem becomes a special and easily solved case of an LP called an "assignment" problem. One of the fastest assignment algorithms is by Gabow and Tarjan [3] with run times roughly proportional to $(n^{1/2})*m*logn$ where $n=(N+M)$ and $m=N*M+n$. When strict inequality of (4a) holds, the assignment algorithm is not valid, but a more general LP code will still be applicable. Only (4) is linear; problems (2) and (3) are nonlinear (NLP).

Results Applicable to Model (3)

Classical optimization theory uses differential calculus to determine points of maxima and minima (extrema) for unconstrained and constrained nonlinear functions. In general, consider the problem:

$$\begin{aligned} &\max x_0 = f(\mathbf{X}) \\ &\text{subject to} \\ &g_i(\mathbf{X}) \le 0,\ i = 1, 2, \ldots, m \\ &\mathbf{X} \le \mathbf{0}. \end{aligned}$$

where f and g_i are nonlinear function(s) of the decision vector X.

The Kuhn-Tucker conditions are sufficient for any local maxima (minima) to be a global maximum (minimum) if both f and the g_i are concave (convex). Lloyd and Witsenhausen [4] have shown that although (3) is concave, it unfortunately is NP-Complete, and thus cannot be practically solved for large-sized problems. One approach that has some hope of finding the global maximum to (3) is to find a maxima by some nonlinear method and then to employ some rounding scheme to get the integer solution. Concavity (convexity) is useful in this situation because it provides an upper (lower) bound on the integer solution; such information could be used to determine whether or not to try to get a better (than current) solution by some rounding schemes. We note here that we insist on integer solutions, and thus any solution based on derivatives, i.e., real solutions, must at last be rounded to integers. To our knowledge no generalized rounding scheme exists to yield the optimal integer solution from an optimal continuous solution of a concave (convex) NLP.

Results Applicable to Model (2)

Unfortunately problem (2) is neither strictly concave nor strictly convex. Although a local maximum may be found, the Kuhn-Tucker conditions are no longer sufficient for this maximum to be the global maximum. A simple example shows why (2) is multimodal [5]:

$$f(x,y) = (1-e^{-x})(1-e^{-y}) \text{ for } x,y \ge 0.$$

The separate factors of f are concave, but their product is not concave; the Hessian is explicitly

$$\begin{vmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{vmatrix} = e^{-(x+y)}\,[\,1 - e^{-x} - e^{-y}].$$

This is negative for x,y near the origin (concavity condition), but is positive for larger x,y (convexity condition). A local maximal solution to (2) tells us nothing about the global maximum, and we would still have the problem of rounding to an integer solution.

As a more general case of an NP-Complete problem, and with a multimodal objective function and integer decision variables, model (2) is a problem in combinatorial optimization that requires heuristic algorithms to solve for all but very small problems.

Solution Approach for Model (2)

The remainder of the paper describes our approach to solving the problem associated with model (2) -- the no partial damage model.

Observe in problem (2) that no value is obtained from the defense of a site unless at least one

shot is placed on each object aimed at it. This means total defense of all sites may not be feasible or cost effective. Therefore it is important, as part of the allocation algorithm, to determine what sites will be defended, and what will be left undefended. Having determined what sites to defend, it then becomes necessary to determine the best firing schedule for the shots against the selected threat objects going to the defended targets. This is the basis of our solution approach. We have a set of defended site selection algorithms, and a set of allocation algorithms. Together, they constitute the algorithm suite. Depending on the situation, a site selection/allocation pair is picked. Each of the site selection algorithms can work with each of the allocation algorithms, creating a large number of possible combinations. The algorithms work iteratively -- sites to be defended are selected, then allocations are made, and the results are fed back to the site selection algorithm for another iteration. Each site selection algorithm has a stopping rule for ending the process and reporting an answer.

Two important measures of effectiveness of an algorithm combination are the run times and the solution quality. These differ according to the defensive situation, so another element of the overall solution approach is a switching function that selects the best algorithm combination from the suite for the given situation. The best pair of algorithms gives the highest solution value within the run time allotted.

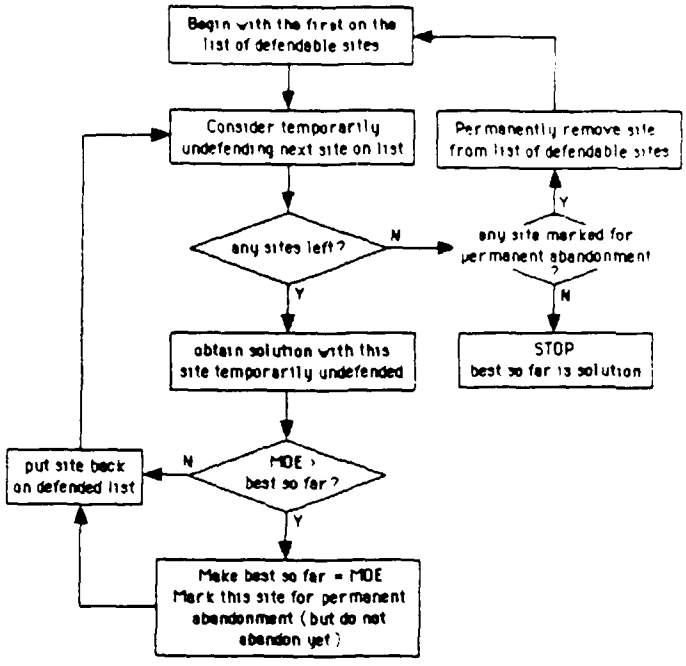
The remaining sections describe in turn each of the specific members of the algorithm suite, first the site selection algorithms, and then the allocation algorithms. Finally, some results are presented that show in which situations each combination is most appropriate. This section is not intended to be a survey of all solution methods, nor is it a complete list of all methods currently under study.

Site Selection Algorithms

Best Branch

Best Branch (BB) is a technique that derives from the Branch and Bound solution approach. Branch and Bound is a standard tool used in integer program solution algorithms. Branch and Bound builds a tree of alternative solutions, then greatly reduces the solution searching by pruning branches whose upper bound for the objective function value is less than some solution already in hand. An upper bound of a branch is not less than any solution from the branch. This approach is a partial enumeration technique, and the effectiveness of the pruning process in reducing the search depends on the tightness of the bounds and the ease of obtaining them.

As a site selection algorithm, BB considers, in turn, the abandonment of each site defended in the current allocation. Unfortunately, we know of no quick, easy ways of getting tight upper bounds for these branches -- instead we figure the actual solution with the undefended sites explicitly eliminated from the allocations. This may not yield an upper bound, nevertheless we use the values obtained from these subproblem solutions to eliminate the corresponding branches from consideration. Specifically, the subproblem objective function values are compared to the best solution found so far, if not as good the site is taken off the undefended list. A block diagram of this heuristic is in Figure 1.



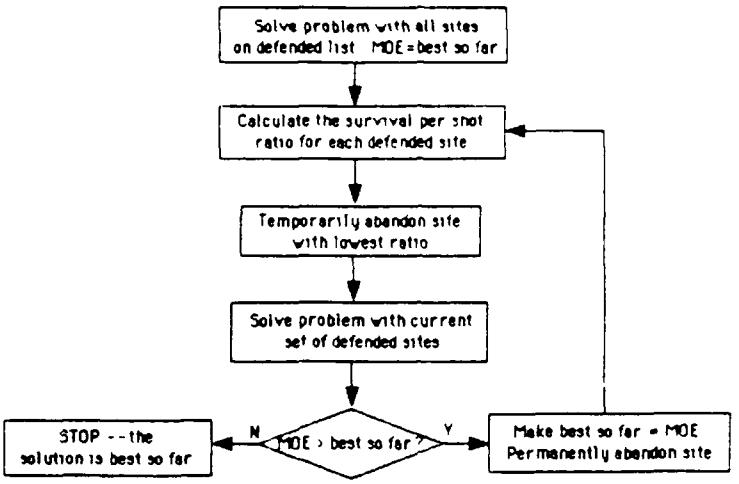
Best Branch Site Selection Heuristic
Figure 1

King of the Hill

Like the game of King-of-the-Hill (KOH) where at each round the last to scramble to the top of the hill is eliminated, this site selection heuristic eliminates at each iteration the site with the poorest shot effectiveness until only the most efficient shots are taken. Initially all sites are considered defendable, and an allocation to defend all sites is attempted. Then the site with the most ineffective shots is eliminated from the defense, and a new allocation is made. The algorithm iterates, dropping sites one at a time from the defense. Each time a site is dropped, the shots allocated to it can be applied elsewhere, possibly more effectively. Finally when the loss of a site's value is not compensated for by the increased effectiveness of its shots elsewhere, the objective function will decrease. The algorithm stops when the expected surviving value is reduced from the previous iteration.

After each iteration of dropping a site and determining a new weapon allocation, the shot efficiency of the shots defending each site must be computed. This is the ratio of the site's expected survivability divided by the number of shots expended to achieve that value.

This heuristic is intended to be fast, with the number of iterations equal to the number of sites only in the worst case where only one site is defended. It does not attempt to reconsider dropped sites to seek better combinations of defensible sites. A block diagram of the KOH heuristic is in Figure 2.



King of the Hill Site Selection Heuristic
Figure 2

Genetic Search

The idea behind the Genetic Search algorithm was first introduced by Holland [7] in 1975. A combinatorial optimization problem may be stated: given a family P of feasible combinatorial structures, each represented by a point p, and a utility $u(p)$ defined over the members of P, find the one of them with maximum (minimum) utility. In a maximization problem only the higher utility points are retained in an "offspring pool" of some fixed size. The n^{th} "generation" of points $p_{.,n}$ creates the next generation of offspring $p_{.,n+1}$ via a set of genetic operators. The *cross-over* genetic operator is the primary operator and mates two defended site vectors to yield two new defended site vectors. (For convenience we refer to these as two "defenses"). Mate selection for cross-over occurs randomly. Consider each defense (a vector of 0s and 1s indexed by site) to be a string with a head at the start of the string and tail at the end. A common point for division between the head and tail may be determined randomly or by some feasibility rules. A cross-over between two defenses occurs by exchanging tails to give two offspring defenses. For example, the following cross-over occurs with a division between positions 5 and 6.

$$\{10001110\} \times \{10000111\} \rightarrow \{10001111\}, \{10000110\}$$

The *inversion* genetic operator occurs after cross-over on a randomly selected segment of an offspring defense. The inversion operation occurs with a specified probability. An example from the preceding cross-over is

$$\{10001111\} \rightarrow \{10010111\}$$

where positions 4 and 5 have been flip-flopped.

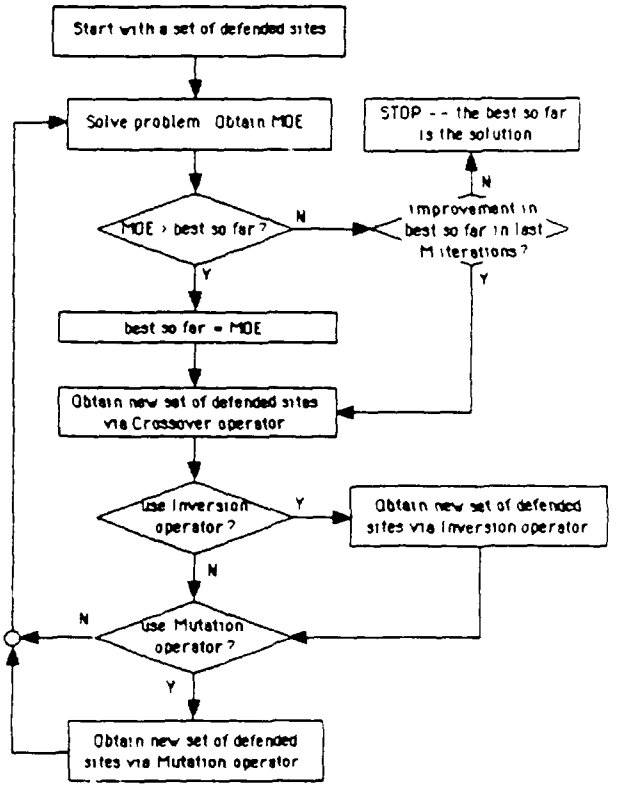
The *mutation* genetic operator alters a single element after an inversion or cross-over operation with another given probability. An example from the preceding inversion is

$$\{10010111\} \rightarrow \{10010101\}$$

where the 1 in position 7 has been changed to a 0. The operations of inversion and mutation occur

rarely and occur principally to prevent the search from getting stuck at local optima. In this way a new generation of feasible defenses is drawn genetically from the current generation.

Each offspring defense is submitted to a weapons allocation algorithm and evaluated by the objective function. As more and more defenses are considered, the chance of finding and retaining better solutions is increased. At some point, though, with very good solutions in hand, the chance of finding better solutions starts to becomes lower. The probabilities of the mutant and inversion operations determine how quickly the process converges and how often the search terminates at a local maximum. The search terminates when some number of redundant best defenses are generated. The genetic search is inherently a parallel algorithm and is applicable where the number of sites is large in comparison to the numbers of shots and objects. A block diagram of this Genetic Search heuristic is in Figure 3.



Genetic Search Site Selection Heuristic
Figure 3

Allocation Algorithms

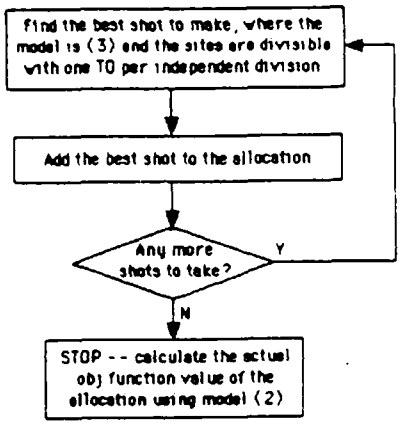
Both of the allocation algorithms are variations of the greedy algorithm. The greedy algorithm is often used in combinatorial heuristics and is based on the idea that at each iteration the best current choice, according to some metric, is chosen from the range of possible alternatives. This choice is myopic in that the selection at a particular iteration may affect the choices in future iterations, but these effects are expressly not considered in the current iteration. For example, this technique might proceed by entering into the solution at each iteration a shot from the weapon/object/time combination with the largest increase in the objective function. The procedure stops when all shots have been expended, or when no more increases can be realized.

Worst case performance bounds on the quality of solution for the greedy have been analyzed by Hausmann, et. al., [6]. Specific bounds are very problem dependent, but for those problems considered the bounds for the worst case performance of the greedy are no less than 50% of the true maximum.

Alias Algorithm

The concept behind Alias is in each greedy iteration to search for an allocation for the vastly simpler problem of model (3), but to insert the resulting solution into the objective function of model (2) when determining the ultimate measure of effectiveness.

The implementation of Alias on the one-threat-per-target problem considers each threat object aimed at a site as going to a separate target which is a fraction of the actual target. (that fraction having value equal to the original target's value divided by the number of objects going to it). The destruction of the fraction does not affect the other fractions. In the end, some threat objects may receive multiple shots while others going to the same site are not shot even once. The total expected surviving values of the sites according to the objective of model (3) would not be the same as that of model (2), so the model (2) objective value of the allocation must be computed using the solution allocation. A block diagram of this allocation heuristic is in Figure 4.



Alias Allocation Heuristic
Figure 4

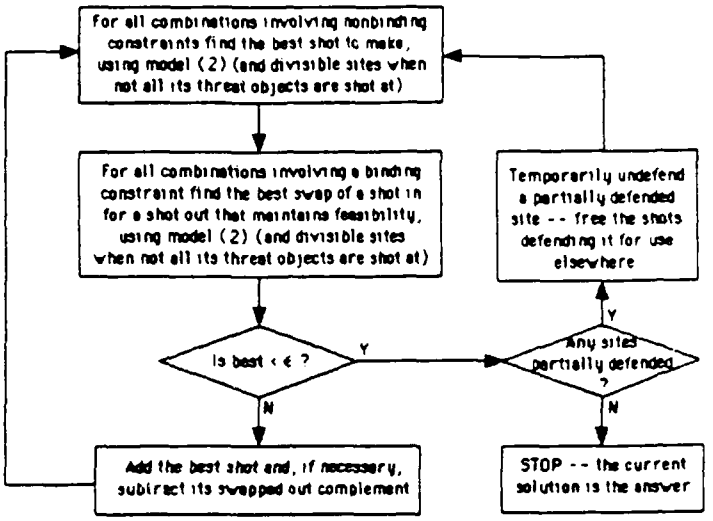
Marginal Algorithm

Like the Alias algorithm, the Marginal allocation algorithm strives to find the best allocation of shots by building up shot by shot. At each step, a weapon-object-time period combination is found where the addition of a shot to the allocation would yield the best marginal increase in the objective function. In this case the objective is the true one from model (2). If the best increase falls below a pre-set minimum the algorithm stops.

The marginal change in the objective function due to a shot is figured simply by determining the value of the function if the extra shot were made and subtracting the value of the function without the shot. If the addition of a shot would violate a constraint (e.g. total numbers of shots, weapon inventory, etc.) an already allocated shot must be eliminated so that the constraint remains satisfied. The marginal change in this case is the difference in the objective function after both the addition of the extra shot and the deletion of the currently allocated shot. Among those that could be eliminated to maintain feasibility, the current shot that achieves the best marginal change in the objective is chosen to be swapped out if the new shot is brought in.

The objective function (2) is such that the expected survivability is zero for targets with multiple objects directed at them (N-object targets) but where not all objects have been shot at, but it jumps

discontinuously to a positive value as soon as the last object receives a shot allocation. Because of this, shots on N-object targets where not all N objects have shot allocated to them are treated, for the purposes of figuring marginals, as shots on single-object targets with weights of w/N. At the end, when the best marginal increase drops below the threshold, there may be some targets that are left partially defended -- some shots are allocated, but fewer than the number of objects. In this case, one of these targets is made temporarily undefended, and the algorithm repeats with the shots on this target being reallocated elsewhere. This outer loop continues until no targets are left partially defended. A block diagram of this heuristic is in Figure 5.

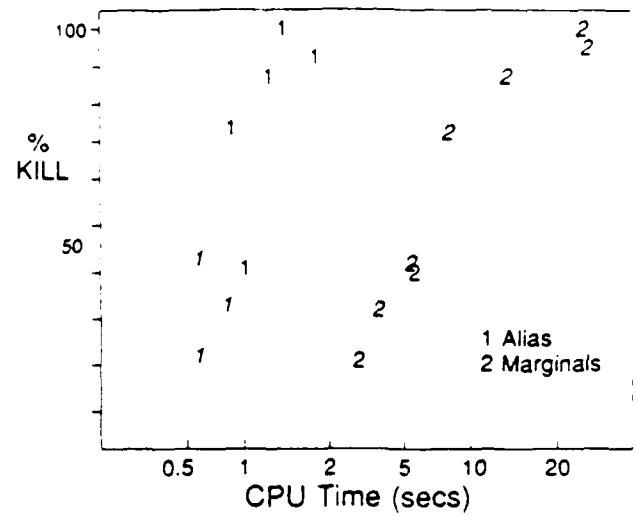


Marginal Allocation Heuristic
Figure 5

Test Results

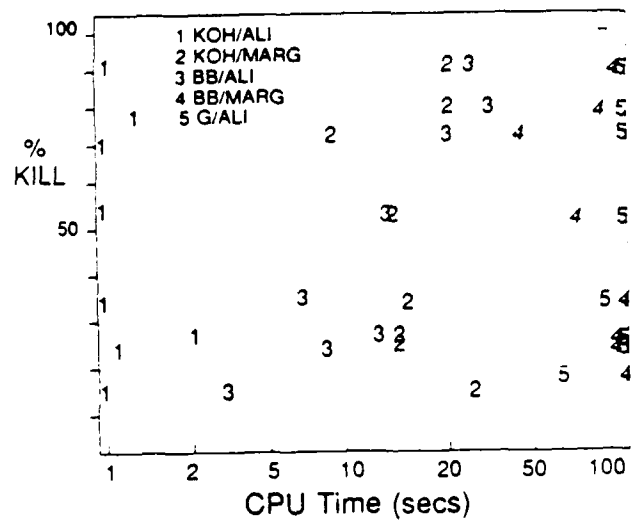
Each of the algorithm combinations was tested for certain defensive situations. These situations were distinguished by varying the numbers of threat objects, their distribution into target groups, the number of shots, and other shot limitations. In particular, the overall shot limits were varied from 25 to 400, while the weapon inventories were either 10, 50, or 100. The number of times each weapon could fire in a time period (refire constraint) was either 1, 5, or 10. The threat was usually 100 objects, but in one test 1000 objects were considered. Two measures of performance were the Performance Ratio (PR), the expected survivability divided by the total values of the targets, and the CPU running time of the algorithms.

The first set of tests was with all targets attacked by single objects. In this case the site selection algorithms were not applicable – the allocation algorithms themselves were sufficient to solve the problem. The run times of Alias were roughly an order of magnitude faster than Marginal. Marginal consistently computed better answers than Alias. The Maximum difference in PR between Marginal and Alias was 3%, and occurred when the number of shots equaled the number of objects. Typical differences were on the order of 0.1%. A graph of the performance of the two algorithms is in Figure 6.



Single Object per Target Case
Figure 6

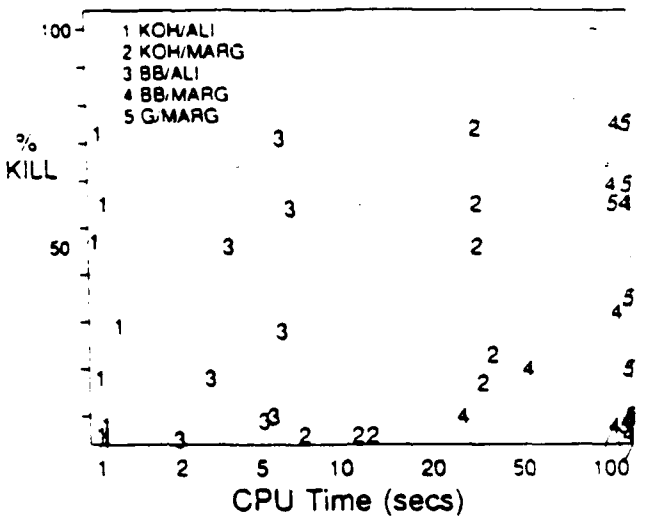
The second set of test cases postulated targets each attacked by small numbers of objects. The Genetic/Marginal combination was not run because its CPU times were too excessive. The combinations invovling Marginal gave better PR values, especially as the numbers of shots per object increased, but were consistently slower than the combinations involving Alias. A graph of the relative performances of the various algorithm combinations is in Figure 7.



Small Numbers of Objects per Target Case
Figure 7

The third set of tests was when targets were each attacked by large numbers of objects. The combination providing the best PR measure was Genetic/Marginal, although BB/Marginal was often within a few percentage points. BB/Alias differed from the maximum by no more than 3% in all cases. As with the second set of tests, combinations involving Marginal were consistently slower, but achieved

better PR measures as the number of shots available increased -- the shot rich cases. A graph of the performances of all algorithm combinations is in Figure 8.



Large Numbers of Objects per Target Case
Figure 8

Summary

The role of weapons allocation for a battle manager in a fairly diverse battle engagement environment after missile launch has been modeled by a set of four problem formulations. The no partial damage model that was chosen for extensive study requires heuristic algorithms for its solution. An iterative solution approach was taken, where in each iteration sites are chosen to be defended, followed by allocations to protect the chosen sites. A suite of site selection and allocation algorithms has been developed, to work in combinations. The benefits of the particular ideas and trade offs embodied in the algorithms of the suite depend, in part, on the defensive situation. The site selection algorithms can generally be ranked according to run time from KOH (fastest), to BB, to Genetic (slowest). The Alias allocation algorithm is always faster than Marginal. The very fast combination KOH/Alias performed within 2% of the best of all combinations 90% of the time; while the slower B&B/Marginal provided the best answer 99% of the time. Work is continuing on refining the current set of algorithms based on testing experience, and on generating entirely new algorithms based on new ideas.

References

[1] W.T. Read Jr., *Tactics and Deployment for Anti-Missile Defense*, Bell Telephone Laboratories, Whippany, N.J., 1958.

[2] S.A. Burr, J.E. Falk, and A.F. Karr, *Integer Prim-Read Solutions to a Class of Target Defense Problems*, Operations Research, V33/4, August, 1985.

[3] Gabow and Tarjan, AT&T Bell Laboratories, Whippany, N.J., 1986.

[4] S.P. Lloyd and H.S. Witsenhausen, *Weapons Allocation is NP-Complete*, IEEE Summer Simulation Conference, 1986.

[5] S.P. Lloyd, Personal communication, AT&T Bell Laboratories, Whippany, N.J., March 1987.

[6] D. Hausmann, B. Korte, and T.A. Jenkyns, "Worst Case Analysis of Greedy Type Algorithms for Independence Systems," *Mathematical Programming Study*, 12, North-Holland, New York, 1980, 120-131.

[7] J.H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.