October, 1990 LIDS-P 2003

Research Supported By:

ONR contract N00014-85-K-0782
ONR contract N00014-84-K-0519

Preferential Defense Strategies. Part II: The Dynamic Case

Hosein, P.A.
Athans, M.

LIDS-P-2003

Preferential Defense Strategies. Part II: The Dynamic Case *

Patrick A Hosein† Michael Athans‡

October 1, 1990

Abstract

In a military conflict, the defense tries to save as many of its assets as possible. In this paper we will assume that the defense can fire its weapons in stages and that the outcomes of the engagements of a stage are observed before assignments for the next stage are made. We will outline a method for deciding on the number of weapons to be used in each stage and the assignment of these weapons.

1 Introduction

In part I of this report [1] we considered the static version of the Asset-Based Weapon-Target Assignment (WTA) problem. In this paper we will consider a dynamic version of the problem. In the dynamic version of the problem we will assume that the defense can fire its weapons in stages and that the outcomes of the engagements of each stage can be used in making decisions for the remaining stages. The defense is said to have a "shoot-look-shoot" capability. The dynamic problem is extremely difficult and so simplifying assumptions are needed to reduce the complexity so as to derive efficient heuristics.

We will make the assumption that the kill probability [1] of a weapon-target pair and the lethality probability of a target-asset pair depend solely on the asset to which the target is directed. Under these assumptions the number of decision variables per stage equals the number of assets. Without these assumptions, the number of decision variables per stage equals the product of the number of available weapons and the number of surviving targets. Therefore, the assumptions greatly reduce the dimensionality of the problem. These are restrictive assumptions which will be violated in most

*This research was conducted at the MIT Laboratory for Information and Decision Systems with partial support provided by the Joint Directors of Laboratories under contract ONR/N00014-85-K-0782 and by the Office of Naval Research under contract ONR/N00014-84-K-0519.

†AT&T Bell Laboratories, Holmdel, New Jersey 07733.

‡Massachusetts Institute of Technology, Cambridge, Massachusetts 02139.

[1] We will use the terminology that was defined in Part I of this report.

practical problems. However, we believe that the general problem will have properties similar to those of our simplified problem.

In section 2 we will define the general problem and discuss its complexity. In section 3 we will give a mathematical statement of the problem under the assumptions of asset dependent kill and lethality probabilities. Because of the extreme complexity of the problem, we will only consider the case of two stages. We will show that, under the assumptions made, the decision variables are the number of weapons to be used in the first stage and the optimal assignment of these weapons. In section 4 we will discuss the problem of finding the optimal number of weapons to be used in stage 1. We will find that this is a difficult problem because of the presence of multiple local optima. In section 5 we will assume that the optimal number of weapons to be used in the first stage is known and discuss the problem of finding the optimal assignment of these weapons. We will present a sub-optimal algorithm for this problem. In section 6 we will present several numerical results. We will find that, in general, a dynamic strategy outperforms a static one by a factor of two. Finally in section 7 we will make some concluding remarks.

2 Problem Definition

This problem consists of a number of time stages. In each stage the results (survival or destruction of each target) of the engagements of the previous stage are observed. Based on these observations, a subset of the remaining weapons is chosen and assigned to the surviving targets. The results of the engagements of this assignment is then observed and the process is repeated. Hence we are dealing with a "shoot-look-shoot-..." strategy. The objective is to choose and assign weapons at each stage so as to maximize the total expected value of the surviving assets at the end of the final stage of the engagement. Note that the problem will be re-solved after each stage because the results of that stage can be observed. This means that one is only interested in obtaining assignments for the present stage. By the principle of optimality, it is implicitly assumed that optimal assignments will be used in all subsequent stages.

We will first define the general problem. In the next section we will consider the special case of two stages under the assumptions that the kill probability of a weapon-target pair depends solely on the asset to which the target is directed, and the lethality probability of a target-asset pair depends solely on the asset. The following notation will be used.

$K \stackrel{\text{def}}{=}$ the number of defense assets,

$T \stackrel{\text{def}}{=}$ the number of time stages,

$N \stackrel{\text{def}}{=}$ the initial number of targets,

$M \stackrel{\text{def}}{=}$ the total number of weapons,

$W_k \stackrel{\text{def}}{=}$ the value of asset k, $\quad k = 1,2,\ldots,K$,

$G_k \stackrel{\text{def}}{=}$ the set of targets aimed for asset k initially, $k = 1,2,\ldots,K$,

$n_k(t) \stackrel{\text{def}}{=}$ the number of targets aimed for asset k in stage t, $k = 1,2,\ldots,K$,

$p_{ij}(t) \stackrel{\text{def}}{=}$ the probability that weapon j destroys target i in stage t if assigned to it,

$i = 1,1\ldots,N, \quad j = 1,2,\ldots,M$,

$\pi_i \stackrel{\text{def}}{=}$ the lethality probability of target i on the asset to which it is aimed,

$i = 1,2,\ldots,N$.

The decision variables will be denoted by:

$$x_{ij} = \begin{cases} 1 & \text{if weapon } j \text{ is assigned to target } i \text{ in stage 1} \\ 0 & \text{otherwise} \end{cases}$$

Note that we only need to solve for the decision variables in stage 1. The decision variables for all subsequent stages will be obtained after the outcomes of the weapon-target engagements of the previous stage is observed.

The *target state* of the system at the end of the first stage will be defined as the set of surviving targets. This state will be denoted by an N- dimensional binary vector $\vec{u} \in \{0,1\}^N$ and represented by

$$u_i = \begin{cases} 1 & \text{if target } i \text{ survives stage 1} \\ 0 & \text{if target } i \text{ is destroyed in stage 1.} \end{cases}$$

The *weapon state* of the system at the end of stage one will be defined as the set of available weapons. This state will be denoted by an M-dimensional binary vector $\vec{w} \in \{0,1\}^M$ and represented by:

$$w_j = \begin{cases} 1 & \text{if weapon } j \text{ was not used in stage 1,} \\ 0 & \text{if weapon } j \text{ was used in stage 1.} \end{cases}$$

The target state evolves stochastically. The stochastic evolution of the target state in stage 1 depends on the assignment decisions made in stage 1. Given a first stage assignment of $\{x_{ij}\}$, the state at the start of the second stage is an N-dimensional random vector. The probability that u_i is 1 is the probability that target i survives the first stage. The probability that u_i is 0 is the

probability that target i is destroyed in the first stage. The distribution of the random variable u_i is therefore given by:

$$\Pr[u_i = k] = k \prod_{j=1}^{M} (1 - p_{ij}(1))^{x_{ij}} + [1 - k] \left\{ 1 - \prod_{j=1}^{M} (1 - p_{ij}(1))^{x_{ij}} \right\}, \tag{1}$$

$$\text{for} \qquad k = 0, 1, \qquad i = 1, 2, \ldots, N.$$

Equation 1 will be called the *target state evolution* of the system.

The evolution of the weapon state is deterministic and depends on the assignments made in the first stage. The evolution is given by:

$$w_j = 1 - \sum_{i=1}^{N} x_{ij}, \qquad j = 1, 2, \ldots, M. \tag{2}$$

This simply says that weapon j is available in the second stage if and only if it is not used in the first stage. Equation 2 will be called the *weapon state evolution* of the system.

We will let $J_2^*(\vec{u}, \vec{w})$ denote the *optimal* value of a $T-1$ stage problem in which the initial target state is $\vec{u}$ and the initial weapon state $\vec{w}$. This problem has the same form as the T stage problem which is being defined. The $T-1$ stage problem can be defined in terms of the optimal values of $T-2$ stage problems etc. The $T-(T-1)$ or single-stage problem can be defined in terms of the optimal values of 0-stage problems. If the target state at the end of the final stage is $\vec{u}$ and the weapon state at the end of the final stage is $\vec{w}$ (which would be $[0, \ldots, 0]$ for an optimal strategy) then the optimal value of the 0-stage problem is given by

$$J_T^*(\vec{u}, \vec{w}) = \sum_{k=1}^{K} W_k \prod_{i \in G_k} (1 - \pi_i u_i).$$

In other words this is the value if no more weapons are assigned and, targets which have been destroyed have a lethality probability of 0 while each target i which survived all stages has a lethality probability of π_i. We can now state the problem as follows.

Problem 2.1 *The Dynamic Asset-Based problem (DAB) can be stated as:*

$$\min_{\{x_{ij}\}} J_1 = \sum_{\vec{\omega} \in \{0,1\}^N} \Pr[\vec{u} = \vec{\omega}] J_2^*(\vec{\omega}, \vec{w})$$

$$\text{subject to} \quad x_{ij} \in \{0, 1\}, \qquad i = 1, 2, \ldots, N \quad j = 1, 2, \ldots, M,$$

$$\text{with} \quad w_j = 1 - \sum_{i=1}^{N} x_{ij}.$$

The objective function is the sum over all possible stage 2 target states of the probability of occurrence of that state times the optimal value given that state. Note that the distribution of the stage 2 target state and the stage 2 weapon state both depend on the first stage assignment. The first constraint restricts each weapon to be assigned at most once in the first stage. The second constraint is due to the weapon state evolution.

This problem is considerably more difficult than the static one. Note that to simply evaluate the expected value of a first stage assignment requires a tremendous computational effort. Besides the problem of dimensionality there is also the difficulty of solving the static problem in the last stage. Several of these static problems must be solved corresponding to the different possible outcomes. Since there are no efficient algorithms for obtaining the optimal value of the static problem, we cannot even evaluate an arbitrary assignment for the dynamic problem. These difficulties have forced us to make some simplifying assumptions. We believe that this simplified problem will reflect the overall behaviour of the more general problem.

3 The Two-Stage Case with Asset Dependent Kill and Lethality Probabilities

Because of the tremendous complexity of the general version of the problem we will make some simplifying assumptions. We will only consider the case of two stages since the complexity of the problem grows exponentially with the number of stages. We will make the assumption that the kill probability of a weapon-target pair depends solely on the asset to which the target is directed. Therefore the kill probability of any weapon on a target aimed for asset k will be denoted by p_k. We will also assume that the lethality probability of a target-asset pair depends solely on the asset. Therefore the lethality probability of each of the targets aimed for asset k will be denoted by π_k.

Because of the assumption of weapon independent kill probabilities, we can let the decision variables be the number of weapons assigned to each target in each stage. Furthermore the assumptions imply that all targets directed to a specific asset are identical. Therefore, in each stage, the number of weapons asigned to any two targets aimed for the same asset cannot differ by more than one. In other words the weapons assigned to defend an asset in a stage must be spread as evenly as possible among the surviving targets aimed for that asset. This result can be used to simplify the problem even further by defining the decision variables as the number of weapons

assigned to defend each asset in each stage. We can therefore let the decision variables be m_1, the number of weapons to be used in stage one, and $\vec{X} \in Z_+^K$ the assignment of these m_1 weapons in stage 1, where X_k represents the number of weapons assigned to defend asset k in stage one. The individual target assignments can be obtained by spreading these weapons as evenly as possible among the targets aimed for asset k.

Our assumptions can also be used to simplify the representation of the target state. Since all targets directed to a specific asset are identical then we can represent the target state by $\vec{n}(2)$ where $n_k(2)$ is the number of targets aimed for asset k that survive the first stage.

The state $\vec{n}(t)$ of the system evolves stochastically. This evolution depends on the weapon assignments made. Because we assumed that the engagement of a target by a weapon in a stage is independent of all other engagements in all stages then, given an assignment for the first stage, the state $n_k(2)$ of asset k evolves independently of all other assets. The state for each asset evolves as follows. To simplify the expression we have left out the subscript k from the variables $n_k(\cdot), p_k(\cdot), q_k(\cdot)$. [2]

$$\begin{aligned}
&\Pr[n(2) = j | X = \chi] = \qquad (3)\\
&= \sum_{\ell=\underline{\ell}}^{\bar{\ell}} \binom{\chi - n(1)\lfloor \frac{\chi}{n(1)} \rfloor}{\ell} q(1)^{\ell \lceil \frac{\chi}{n(1)} \rceil} [1 - q(1)^{\lceil \frac{\chi}{n(1)} \rceil}]^{\chi - n(1)\lfloor \frac{\chi}{n(1)} \rfloor - \ell} \times \\
&\quad \times \binom{n(1)\lfloor \frac{\chi}{n(1)} \rfloor + n(1) - \chi}{j - \ell} q(1)^{(j-\ell)\lfloor \frac{\chi}{n(1)} \rfloor} [1 - q(1)^{\lfloor \frac{\chi}{n(1)} \rfloor}]^{n(1)\lfloor \frac{\chi}{n(1)} \rfloor + n(1) + \ell - \chi - j}
\end{aligned}$$

for

$$j = 0, 1, \ldots, n(1).$$

where

$$\underline{\ell} = \max\{j + \chi - n(1)(\left\lfloor \frac{\chi}{n(1)} \right\rfloor + 1), 0\} \quad \text{and} \quad \bar{\ell} = \min\{\chi - n(1) \left\lfloor \frac{\chi}{n(1)} \right\rfloor, j\}$$

This evolution can be explained as follows. If X_k is a multiple of $n_k(1)$ then $n_k(2)$ is a binomial random variable with success probability $(1 - p_k(1))^{\frac{X_k}{n_k(1)}}$. If X_k is not a multiple of $n_k(1)$, then some targets will be assigned $\lfloor \frac{X_k}{n_k(1)} \rfloor$ weapons while the others will be assigned $\lceil \frac{X_k}{n_k(1)} \rceil$ weapons. The distribution of the random variable $n_k(2)$ is obtained by convolving two binomial distributions. The success probability of one of these distributions is given by $(1 - p(1))^{\lfloor \frac{\chi}{n(1)} \rfloor}$, while the success

[2] For ane real variable x, $\lceil x \rceil$ denotes the smallest integer greater than or equal to x while $\lfloor x \rfloor$ denotes the largest integer less than or equal to x.

probability of the other is given by $(1-p(1))^{\lceil \frac{X}{n(1)} \rceil}$. The variables $\underline{\ell}$ and $\bar{\ell}$ were introduced to take care of the boundary conditions of the convolution.

Let $J_2^*(\vec{n}(2), M)$ denote the optimal value of the second stage problem with target state $\vec{n}(2)$ and M weapons. Also let $\mathcal{S}$ denote the set of all possible outcomes of the first stage

$$\mathcal{S} = \{\vec{s} \in Z_+^K | s_k \in \{0, 1, \ldots, n_k(1)\}\}.$$

We will state the two-stage problem in terms of the optimal values of single stage problems. The single-stage problem is simply a static problem. However, we can use the same recursive definition that was used for the two-stage problem to define the single-stage problem in terms of optimal values of 0-stage problems. Note that in the optimal strategy no weapons will be available after the second stage. Denote the target state after stage two, the final stage, by $\vec{n}(3)$. The optimal value of the 0-stage problem is given by:

$$J_3^*(\vec{n}(3), 0) = \sum_{k=1}^{K} W_k (1-\pi_k)^{n_k(0)}.$$

In other words J_3^* is the total expected value of the surviving assets if the target state is $\vec{n}(3)$ and no more weapons are fired.

Problem 3.1 *The Two-Stage, Dynamic, Asset-Based (TDAB) problem with asset dependent kill and lethality probabilities can be stated as:*

$$\begin{aligned} \max_{\{\vec{X}\}} J_1 &= \sum_{\vec{s} \in \mathcal{S}} \Pr[\vec{n}(2) = \vec{s}] J_2(\vec{s}, M - m_1) \\ \text{subject to} \quad & X_k \in Z_+, \qquad k = 1, \ldots, K \\ & \text{and} \quad \sum_{k=1}^{K} X_k = m_1. \end{aligned}$$

One can see that even the statement of the problem is a formidable task even under the assumption that the kill and lethality probabilities are solely asset dependent.

By the principle of optimality, the assignments used in the second stage must be optimal. Therefore, the only decision variables over which the objective function is to be optimized are m_1 and $\vec{X}$, which is the number of weapons to be used in the first stage m_1 and the assignment of these weapons to assets $\vec{X}$. We will therefore denote the optimal value for the case in which m_1 weapons are used in the first stage with assignment $\vec{X}$ by $J_1(m_1, \vec{X})$.

Problem 3.2 *The Dynamic Asset-Based problem may also be stated as:*

$$\max_{m_1 \in Z_+} \left\{ \max_{\vec{X} \in Z_+^K} J_1(m_1, \vec{X}) \right\}$$

$$\text{subject to} \quad \sum_{k=1}^{K} X_k = m_1,$$

$$\text{and} \quad 0 \leq m_1 \leq M.$$

If we fix m_1 then the inner subproblem can be written as

Problem 3.3 *(Assignment subproblem):*

$$\max_{\{\vec{X} \in Z_+^K\}} J_1(m_1, \vec{X})$$

$$\text{subject to} \quad \sum_{k=1}^{K} X_k = m_1.$$

If we can solve the assignment subproblem, then the original problem can be solved as follows. Let $\vec{X}^*$ denote the optimal assignment of the subproblem 3.3. Note that this optimal assignment depends on the value of m_1. However, this value is implicit in the solution since $\sum_{k=1}^{K} X_k^* = m_1$. The solution to the original problem may now be obtained by solving the following:

Problem 3.4 *(Main problem):*

$$\max_{m_1 \in Z_+} J_1(m_1, \vec{X}^*)$$

$$\text{subject to} \quad 0 \leq m_1 \leq M.$$

Each of the problems 3.3 and 3.4 will be considered separately. Our efforts will be concentrated on the solution of problem 3.3 since we will show that problem 3.4 has many maxima and hence, in general, a global search will have to be done to obtain the optimal solution.

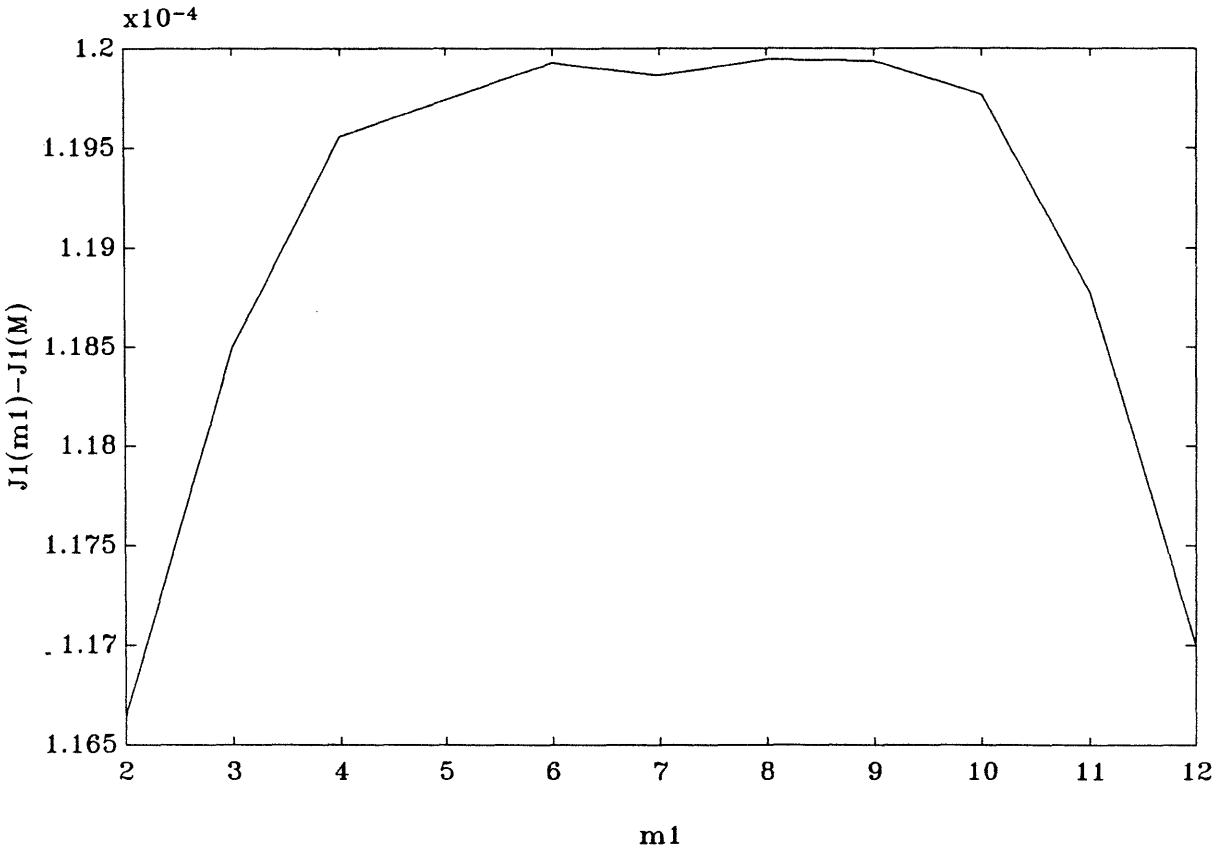

Figure 1: An example of the two-stage dynamic asset-based problem for which multiple maxima exists. Plot of the expected two-stage value $J_1(m_1)$ minus the static value $J_1(M)$ vs. the number of weapons used in stage 1, m_1, with $M = 14, K = 3, \vec{n}(1) = [1,1,1], p = 0.9$.

4 Optimal Number of First-Stage Weapons

Let us assume that we can solve the assignment subproblem 3.3 and consider the problem 3.4 for the case of $T = 2$. Consider, for example, the case $M = 14, K = 3, \vec{n} = [1,1,1]$ and $p_k(t) = 0.9$.

In figure 1 we have plotted $J_1(m_1, \vec{X}^*(m_1)) - J_1(M)$ versus m_1 for this problem. The optimal value of the static strategy was subtracted from that of the dynamic strategy to obtain a scale on which the different maxima are visible. Furthermore, we have only plotted the cases $m_1 = [2, \ldots, 12]$. Again this was done so that the different maxima will be visible. The difference in value of the local maxima is so small that for all practical purposes the solution for any of them will be satisfactory.

Therefore, to obtain the global maximum one must essentially do a global search. For most practical purposes however, a local maximum will suffice. To obtain a local maximum a simple local search algorithm can be used. If several processors are available then the local search algorithm

can be run on each of them simultaneously with different initial solutions. The best local maximum may then be taken.

Problem 3.2 is an important one since it is used to determine the optimal number of weapons to use in the present stage. However, it is also a difficult problem to solve because the objective function is not unimodal. Our belief is that in practice any local maxima will suffice since we conjecture that the difference in the values of any two local maxima will be negligible compared to the value of any one of them. The reason why all solutions cannot be checked is because of the computational requirements for evaluating each solution. This computation can be reduced by making good approximations.

5 Optimal Assignment of the First-Stage Weapons

In this section we will consider the assignment subproblem 3.3. In this problem the number of weapons to be used in the first stage is fixed and the objective is to assign these weapons optimally. Note that for the static version (Part I) of this problem we were able to obtain a suboptimal algorithm but not an optimal one. In this section we will provide a suboptimal algorithm as well. This algorithm is similar to that used to solve the static problem in that it approximates the objective function by a concave, separable one. We will illustrate the algorithm for the case of two stages.

Since there are only two stages then $m_2 = M - m_1$. Let $J_1(\vec{X})$ denote the expected value for a first stage assignment of $\vec{X}$, (with $\sum_{k=1}^{K} X_k \leq m_1$), and m_2 weapons are assigned in the second stage optimally. The function $J_1(\vec{X})$ is non-separable (with respect to the assets) and non-concave. We will approximate this function by a function $\tilde{J}_1(\vec{X})$ which is both separable and concave.

Let e_k denote the k^{th} column of the K-dimensional identity matrix and let X_k be a non-negative integer. Consider the one-dimensional function $J_1(X_k e_k)$. This is the expected value if, in the first stage, X_k weapons are assigned to asset k and no other weapons are assigned in this stage while in the second stage m_2 weapons are assigned optimally. An example of this function is given in figure 2 (the solid line). For this example we used $K = 2, k = 1, \vec{n}(1) = [10, 10], \vec{W} = [1, 1]$, and $\vec{p}(t) = [.4, .4]$. The number of weapons used in stage 2 was fixed at 20.

Note that, as a function of multiples of n, the function is convex and then becomes concave. This property was observed for the static problem as well. However note that, between multiples

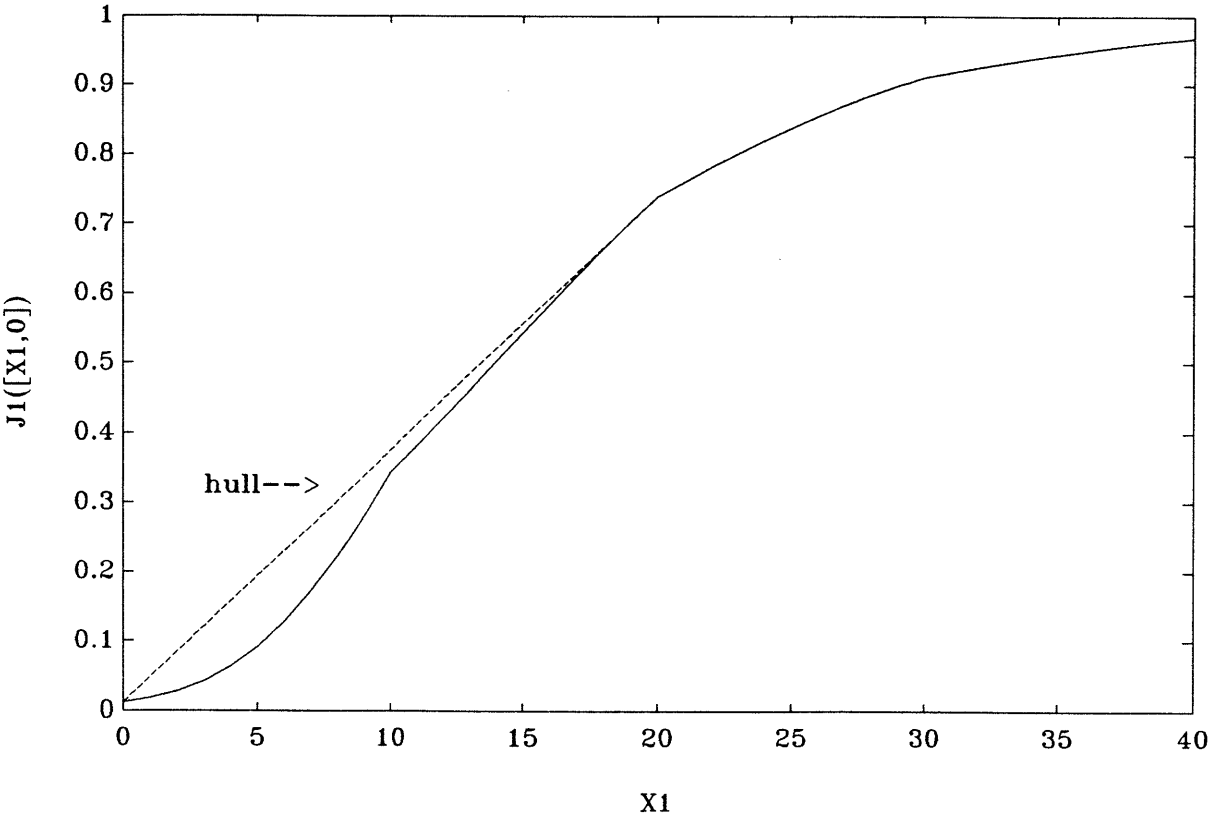

Figure 2: An example the expected two-stage value, $J_1(\vec{X})$, plotted along a coordinate direction for a two-asset problem.

of n the function is convex for small X and concave for larger values of X. This is unlike the static case for which the function was always convex between multiples of n. The reason for this is that, even if only a subset of the targets aimed for an asset are engaged, there is still a significant increase in value because the remaining targets will be engaged in the second stage. We have also included the concave hull (the dashed line) of the function in the plot. We will denote the concave hull of this function by $\tilde{J}_1(X_k e_k)$. Note that the concave hull is a very good approximation to the function.

Let us denote the K-dimensional zero vector by $\vec{0}$. The approximation to the function $J_1(\vec{X})$ which we will use is given by:

$$\tilde{J}_1(\vec{X}) \equiv J_1(\vec{0}) + \sum_{k=1}^{K} [\tilde{J}_1(X_k e_k) - J_1(\vec{0})] \tag{4}$$

where the function $\tilde{J}_1(X_k e_k)$ is the concave hull of the function $J_1(\vec{X})$ along the k^{th} coordinate direction. Along the coordinate directions this approximation is the concave hull as given in figure

2. The values for the interior points are obtained by summing the increases along each coordinate and adding the value at the origin. Note that the value at the origin is the optimal value of the corresponding static problem with m_2 weapons since no weapons are used in stage 1.

Note that the function $\tilde{J}_1(\vec{X})$ is concave and separable with respect to the assets. Furthermore, note that if $m_1 = M$, (i.e. all weapons are used in the first stage) then the problem is a static one and the approximation used is the same as the approximation that was used in the suboptimal algorithm for the static problem that was presented in Part I. Also note that if only enough weapons are used in stage 1 to defend one of the assets, then the approximation is the same as the exact function because along the coordinate directions through the origin both functions are equal in the region in which the asset is defended. Therefore, in the limits of small and large values of m_1 the approximation is good.

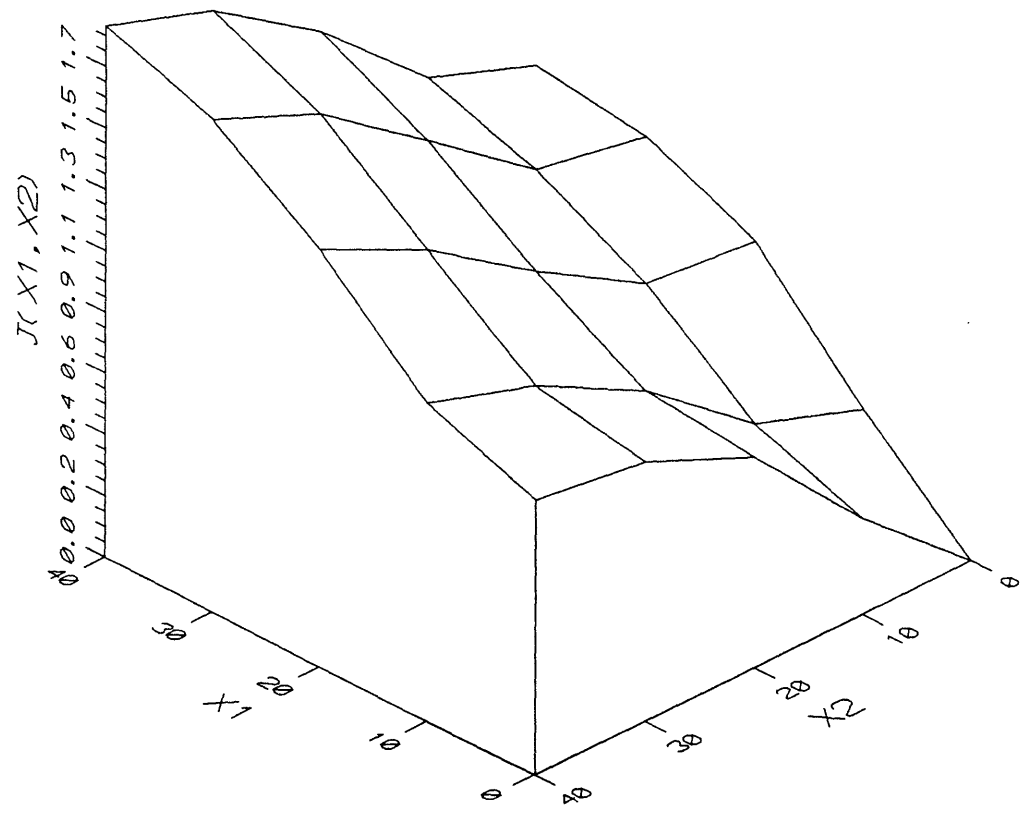

Figure 3: The expected two-stage value $J_1(\vec{X})$, if X_k weapons are assigned to defend asset k in stage 1 and 20 weapons are reserved for stage 2 with $K = 2, n_k(1) = 10, W_k = 1, p_k(t) = .4, \pi_k = 1$ for $k = 1, 2$.

In figure 3 we have plotted the function $J_1(\vec{X})$ versus X_1 and X_2 for the example used for figure

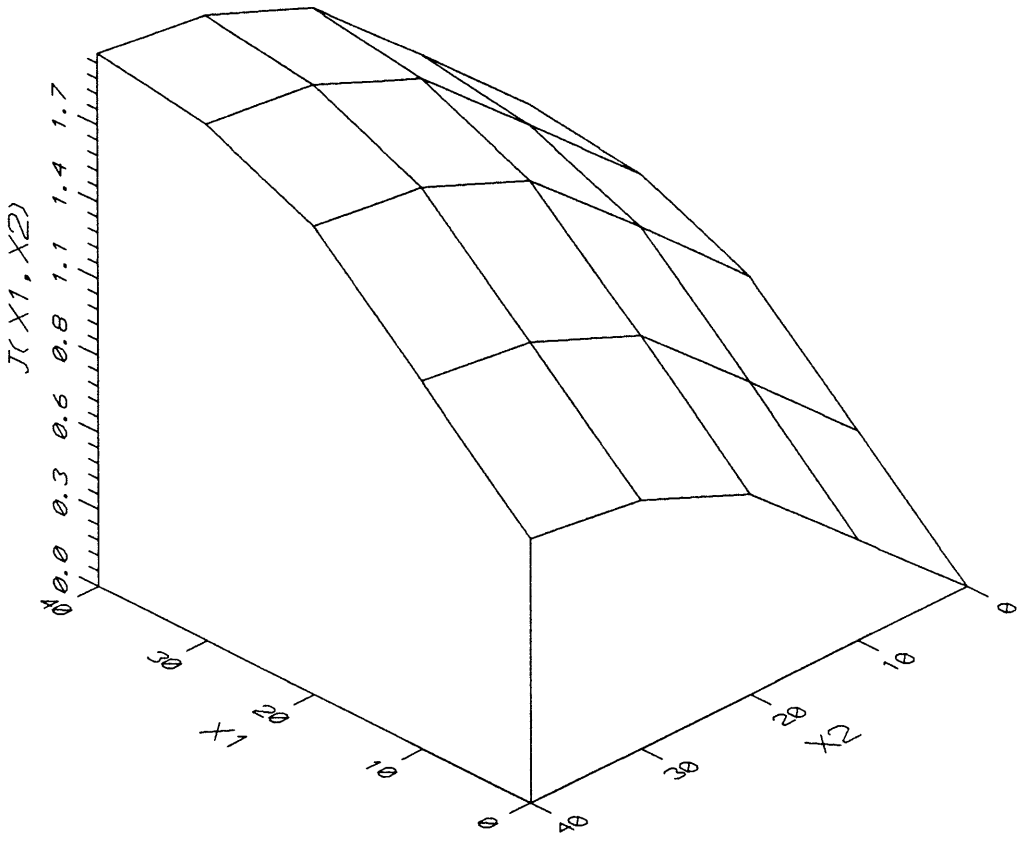

Figure 4: The concave hull approximation, $\tilde{J}_1(\vec{X})$, of the function $J_1(\vec{X})$ given in figure 3.

2. In figure 4 we have plotted the corresponding approximation $\tilde{J}_1(\vec{X})$. We only evaluated the functions at points where X_1 and X_2 were multiples of n. Note that the approximation is good if $X_1 \geq 20$ and $X_2 = 0$ or if $X_2 \geq 20$ and $X_1 = 0$. This is where the solution will lie if only one of the assets is defended. The approximation is also good in the region $X_1 \geq 20, X_2 \geq 20$. This is where the solution will lie if both assets are defended. Also note that the approximation is an upper bound on the true function. The algorithm is given in figure 5.

The suboptimal solution is obtained by solving the problem with the approximate function as the objective. The value of this solution is then evaluated using the exact function. However, since the approximate function is an upper bound then if we evaluate the solution using the approximate function then we can obtain an upper bound on the optimal value of the problem.

Theorem 5.1 *The function $\tilde{J}_1(\vec{X})$ defined by equation 4 is an upper bound to the function $J_1(\vec{X})$, i.e*

$$\tilde{J}_1(\vec{X}) \geq J_1(\vec{X}) \qquad \text{for} \quad \vec{X} \in Z_+^K.$$

procedure DAB
begin
Pick a value for m_1;
Compute the approximate function $\tilde{J}_1(\vec{X})$;
Use MMR algorithm to assign the m_1 first stage weapons using
$\tilde{J}_1(\vec{X})$ as the objective function;
This assignment will be the sub-optimal solution for the dynamic problem;
Evaluate value of assignment using simulations;
end

Figure 5: Algorithm for the Dynamic Asset-Based problem

Proof: The proof of this theorem may be found in the thesis by Hosein[2]. ■

Note that evaluation of any feasible assignment in stage 1 requires an optimal algorithm to compute the optimal stage 2 value for each possible outcome of stage 1. Since we do not have an optimal static algorithm we can only compute a lower bound on the expected value of the solution of the dynamic algorithm. This is done by using the value of the solution produced by the algorithm described in Part I for the solution of the static problem in stage 2. There is also the problem that the number of possible outcomes is enormous. To overcome this problem we use Monte Carlo simulations. We simulate the first stage outcome and then compute the value given that outcome. Several of the simulations are run and the sample mean is taken as an approximation of the value. These simulations will be discussed in detail later.

An upper bound on the optimal value is obtained as follows. Solve the problem in which the objective (dynamic case) function is replaced by the approximate function $\tilde{J}$. We also need to use an upper bound for the value in the second stage. This can be obtained from the sub-optimal algorithm that was presented for the static problem.

6 Numerical Results

In this section we will present several computational results for the Dynamic Asset-Based WTA problem. We will use the algorithm given in figure 5 to solve the problem and will also provide an upper bound on the optimal value for the problem. Some sensitivity analysis results will also be presented.

The following problem will be used as our baseline problem. We will consider the case of two

time stages. The kill probability of each weapon-target pair in each of the stages is $p_k(t) = 0.6$. There are $K = 10$ assets to each of which is aimed $n_k = 10$ targets. The defense has $M = 200$ weapons to intercept these 100 targets. The lethality probability π_k of each target is unity. The value W_k of each of the assets is unity. This problem was chosen as the baseline problem because it illustrates the following. The optimal static strategy of this problem is to defend 5 of the 10 assets. However, we will find that in a dynamic scenario it is better to defend nine of the assets in the first stage. Therefore the number of assets defended is almost doubled if a dynamic strategy is used rather than a static one.

6.1 Discussion of Simulations

As was mentioned in the previous section, our proposed algorithm produces a sub-optimal solution. In order to compute the expected value for this solution one needs an optimal algorithm for the static problem since for each possible first-stage outcome one must find the optimal value for the corresponding static problem. Since this is not available, we will produce a *lower* bound on the value of this solution. This will be done by using a lower bound on the optimal value for each static problem that must be solved. Another difficulty is the number of possible outcomes that must be examined. For example, suppose that in the baseline problem each of the assets had a different value and that in the first stage a single weapon is assigned to each target. If this is the case then for each asset either 10 of the targets aimed for it may survive or 9, ..., or 0. Therefore since there are 10 assets the total number of possible outcomes of stage 1 is 11^{10}. For each of these outcomes one must calculate the corresponding optimal static value. Such a task is overwhelming. This difficulty is overcome by using Monte Carlo Simulations.

We simulate the first stage of the engagement as follows. Let $\vec{X}$ denote the first stage assignment. Because of the uniformity of the problem, the optimal target assignments can be obtained by spreading the weapons assigned to an asset evenly among the targets aimed for that asset. Let us denote the first stage target assignments by $\vec{x}$. The engagements of the weapons on target i is simulated by flipping a coin. The success probability of the coin is $(1 - p_k)^{x_i}$. If the coin toss is a success then we assume that target i survives the first stage, while if the coin toss is a failure then we assume that target i is destroyed in stage 1. This is repeated for all targets to obtain the target state for the second stage. The expected value of this outcome is then computed (actually

only bounds on the expected value can be computed because we do not have an optimal algorithm for the static problem). Several of these Monte Carlo Simulations are performed (we have used 100 runs) and the sample mean is then taken as an approximation to the expected value of the assignment $\vec{X}$. We have found that after about 100 simulations the first two digits of the sample mean remain constant.

6.2 Discussion of Upper Bound Computation

If we fix the number of weapons to be used in stage 1 then one can obtain an upper bound on the optimal value (for that number of first stage weapons) of the problem by solving the dynamic problem with the upper bound approximation $\tilde{J}$. In order to obtain the global solution of the problem one must search over all possible values of m_1, the number of weapons used in stage 1. In figure 6 we have plotted the lower bound on the value of the solution produced by the algorithm (solid line) as well as an upper bound on the optimal value (dashed line) versus the number of weapons used in stage 1. In order to obtain a lower bound on the optimal value of the problem we must choose the maximum over all values of m_1 of the solid line. To obtain an upper bound on the optimal value we must choose the maximum over all values of m_1 of the dashed line. Unfortunately we find that each of these functions peaks at different points. It is, however, very unlikely that the optimal value of m_1 is obtained at the peak of the upper bound because at that point the lower bound is extremely small. *We will therefore assume that the optimal value of m_1 is the point at which the lower bound on the expected value of the solution of the algorithm peaks.* This is a very reasonable assumption since we believe that the shape of this function represents very closely the shape of the optimal one.

6.3 Numerical Examples

In the following tables we will investigate the affect of changing various parameters of the baseline problem. We will provide the results for both the dynamic and static problems for comparisons. Note that, for the baseline problem there are 10 unit valued assets. Therefore the expected surviving value of the assets cannot be greater than 10.

Problem 1: Baseline Problem

The baseline problem: $M = 200, N = 100, T = 2, K = 10, n_k = 10, p_k = 0.6, \pi_k = 1, W_k = 1$ for

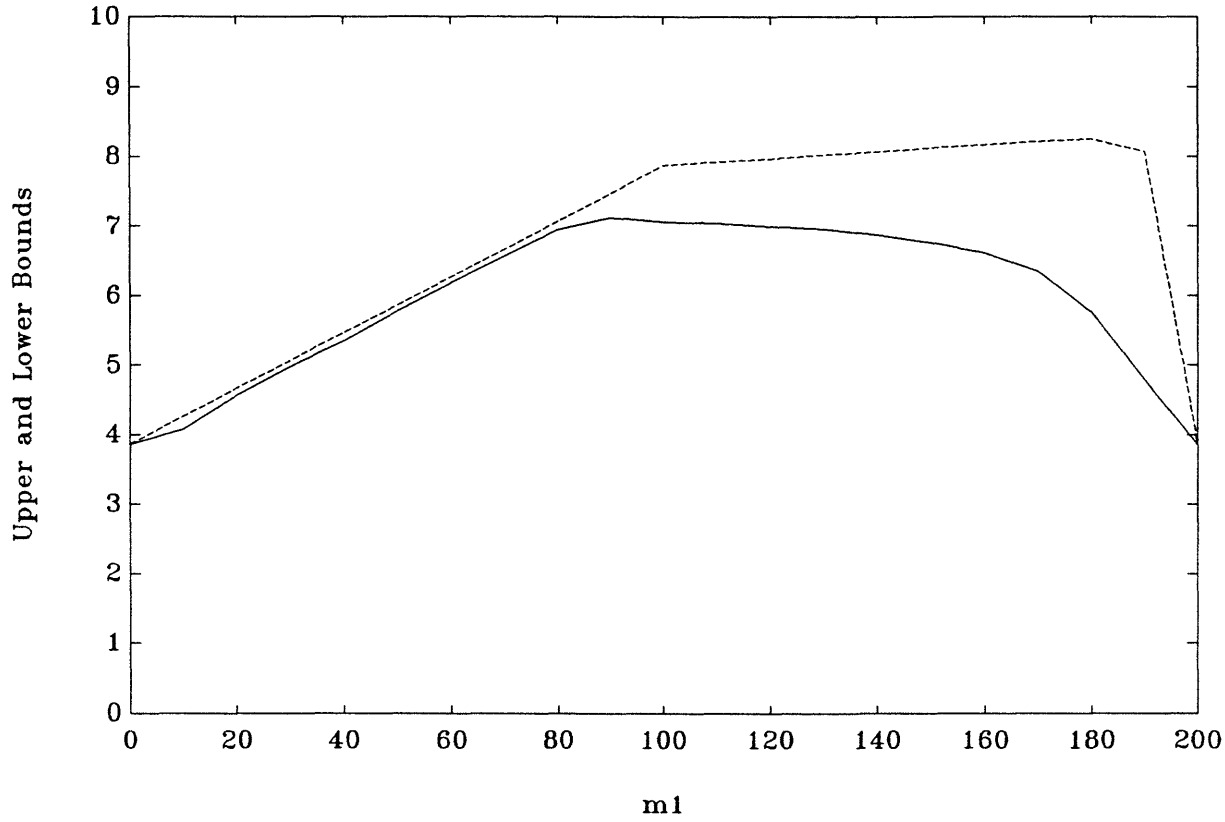

Figure 6: Upper and lower bounds on the optimal value vs. the number of weapons used in stage 1 for the baseline problem.

$k = 1, \ldots, K$.

Static Case:

Optimal solution[3]: [0,0,0,0,0,40,40,40,40,40]

Optimal value: 3.86

Dynamic Case:

Number of weapons used in first stage: 90

Assignment of these weapons[4] : [0,10,10,10,10,10,10,10,10,10]

Lower bound on value of this solution: 7.12

Upper bound on optimal value: 7.46

Remarks:

Note that for the static case the optimal strategy is to defend half of the assets uniformly (prefer-

[3]Represented by the number of weapons assigned to defend each of the 10 assets. The number of weapons assigned to each of the targets directed to an asset can be obtained by dividing by 10 the number of weapons assigned to the defense of that asset.

[4]Represented by the number of weapons assigned to defend each of the 10 assets in the first stage.

ential defense). For the dynamic case 9 of the assets are defended in the first stage. Note also that the value of the solution produced by the sub-optimal algorithm is close to the upper bound on the optimal value. This implies that the sub-optimal solution is either equal or close to the optimal solution.

Note that a typical stage 2 state might be [0,4,4,4,4,4,4,4,4,4]. For this state the optimal stage 2 solution would be [2,12,12,12,12,12,12,12,12,12]. Therefore we see that the same number of assets that were defended in stage 1 are defended in stage 2. This makes sense since weapons would be wasted if more assets were defended in stage 1 than in stage 2.

Finally note that the optimal dynamic value is roughly twice that of the optimal static value. This means that the defense can save roughly twice as many assets by using a dynamic strategy. Since roughly seven assets are eventually saved with the dynamic strategy then why does the defense attempt to save 9 assets in the first stage? Let us consider such a strategy. Consider the assignment in which 160 weapons are used in stage 1. These weapons are used to defend 8 assets with 20 weapons each. The expected value for this solution is 7.09. Therefore we find that if the defense did try to save 8 assets in the first stage then the resulting solution is near-optimal. This suggests that any reasonable strategy will be near-optimal.

Problem 2: Baseline Problem with lower kill probability

The baseline problem except that the kill probability for each weapon in each stage is 0.5:

Static Case:

Optimal solution: [0,0,0,0,0,0,50,50,50,50]

Optimal value: 2.91

Dynamic Case:

Number of weapons used in first stage: 150

Assignment of these weapons: [0,0,10,20,20,20,20,20,20,20]

Lower bound on value of this solution: 5.10

Upper bound on optimal value: 6.90

Remarks:

In this case we note that even for the dynamic problem it is better to use a preferential defense in stage 1. However, 7.5 assets are defended in the dynamic case compared to 4 in the static case.

Upper bound on optimal value: 1.93

Dynamic Case:

Number of weapons used in first stage: 50

Assignment of these weapons: [0,0,0,0,0,10,10,10,10,10]

Lower bound on value of this solution: 3.47

Upper bound on optimal value: 3.93

Remarks:

Here we find that the dynamic strategy performs better than the static one even if the number of weapons equals the number of targets. Again we find that the weapons should be divided equally between the stages. Also note that the performance of the dynamic strategy is approximately twice that of the static one as we have found for most of the problems.

Problem 7: Baseline Problem with more weapons

The baseline problem except that the defense has 300 weapons:

Static Case:

Sub-optimal solution: [0,0,20,40,40,40,40,40,40,40]

Value of suboptimal solution: 5.58

Upper bound on optimal value: 5.79

Dynamic Case:

Number of weapons used in first stage: 200

Assignment of these weapons: [20,20,20,20,20,20,20,20,20,20]

Lower bound on value of this solution: 9.88

Upper bound on optimal value: 10.00

Remarks:

The bound on the optimal value for the dynamic value obtained using our algorithm was actually 10.29. However since there are only 10 assets, each of unit value, the maximum possible value is 10. We therefore find that the algorithm could produce a useless bound as in this case. However we have found that for the cases in which this occurs a good upper bound is the total sum of the asset values. Also note that in this case 200 weapons are used in the first stage. If 150 weapons are used in the first stage the lower bound on the resulting solution is 9.73. Therefore if half of

the weapons are used in the first stage as was the case in most of the other problems the resulting value is still near-optimal.

Problem 8: Baseline Problem with less targets per asset

The baseline problem except that there are 20 assets each of unit value with 5 targets aimed at each asset:

Static Case:

Sub-optimal solution: [0,0,0,0,0,0,5,15,15,...,15]

Value of suboptimal solution: 9.42

Upper bound on optimal value: 9.58

Dynamic Case:

Number of weapons used in first stage: 100

Assignment of these weapons: [5,5,...,5,5]

Lower bound on value of this solution: 16.35

Upper bound on optimal value: 16.61

Remarks:

In this case 82% of the asset value is saved while for the baseline problem 70% was saved This indicates that smaller attacks on each asset favors the defense. This was also true for the static problem. In other words if the number of assets is kept fixed then as the number of targets increases, the performance of the defense decreases even if the weapon to target ratio was kept fixed. Therefore if the defense wishes to maintain the same performance it must increase its arsenal at a greater rate than that of the offense.

Problem 9: Baseline Problem with more assets

The baseline problem except that there are 15 assets of unit value and the defense has 300 weapons:

Static Case:

Sub-optimal solution: [0,0,0,0,0,0,0,0,20,40,40,40,40,40,40,40]

Value of suboptimal solution: 5.58

Upper bound on optimal value: 5.79

Dynamic Case:

Number of weapons used in first stage: 170

Note that the algorithm is able to handle such cases for the dynamic problem.

Problem 3: Baseline Problem with higher kill probability

The baseline problem except that the kill probability for each weapon in each stage is 0.7:

Static Case:

Sub-optimal solution: [0,0,0,20,30,30,30,30,30,30]

Value of suboptimal solution: 4.95

Upper bound on optimal value: 5.07

Dynamic Case:

Number of weapons used in first stage: 100

Assignment of these weapons: [10,10,10,10,10,10,10,10,10,10]

Lower bound on value of this solution: 9.35

Upper bound on optimal value: 9.47

Remarks:

As the kill probability of the weapons increases we find that for the dynamic case all of the assets are defended. Note also that either the solution produced by the algorithm is getting closer to optimal or the upper bound on the optimal value is improving (or both) as the kill probability increases.

Problem 4: Baseline Problem with increasing (with stage) kill probabilities

The baseline problem except that the kill probability of the weapons in the first stage is 0.5 while their kill probability in the second stage is 0.7:

Static Case: (all weapons fired in stage 2)

Sub-optimal solution: [0,0,0,20,30,30,30,30,30,30]

Value of suboptimal solution: 4.95

Upper bound on optimal value: 5.07

Dynamic Case:

Number of weapons used in first stage: 90

Assignment of these weapons: [0,10,10,10,10,10,10,10,10,10]

Lower bound on value of this solution: 6.89

Upper bound on optimal value: 7.22

Remarks:

Our intuition for this case is that more weapons should be used in the stage with higher kill probability (stage 2) than in the other stage. The solution produced by the algorithm does in fact have this property. However, note that although the difference in the kill probabilities is large (0.5 and 0.7) only 20 more weapons are used in stage 2 than in stage 1.

Problem 5: Baseline Problem with decreasing (with stage) kill probabilities

The baseline problem except that the kill probability of the weapons in the first stage is 0.7 while their kill probability in the second stage is 0.5. Note that this is the reverse of problem 4:

Static Case: (all weapons fired in stage 1)

Sub-optimal solution: [0,0,0,20,30,30,30,30,30,30]

Value of suboptimal solution: 4.95

Upper bound on optimal value: 5.07

Dynamic Case:

Number of weapons used in first stage: 120

Assignment of these weapons: [10,10,10,10,10,10,10,10,20,20]

Lower bound on value of this solution: 7.67

Upper bound on optimal value: 8.52

Remarks:

Again note that we obtain the intuitive result that more weapons should be used in the stage with higher kill probability. However, if 100 weapons are used in stage 1 the lower bound on the value of the resulting solution is 7.62. Therefore the value does not seem to be very sensitive to the number of weapons used in stage 1. Finally note that the optimal value for this case is approximately 8.1 while that for the previous problem is approximately 7.1. Therefore we find that it is better to use the more effective weapons in stage 1 rather than in stage 2.

Problem 6: Baseline Problem with less weapons

The baseline problem except that the defense has 100 weapons:

Static Case:

Sub-optimal solution: [0,0,0,0,0,0,0,0,20,40,40]

Value of suboptimal solution: 1.72

Assignment of these weapons: [10,10,10,10,10,10,10,10,10,10,10,10,10,20,20]

Lower bound on value of this solution: 10.57

Upper bound on optimal value: 11.90

Remarks:

In this case we have increased the number of assets while keeping the weapon to target ratio fixed. We find that the percentage of asset value saved in this case (70%) is approximately the same as that for the baseline problem (71%). We also find that the fraction of weapons used in the first stage is closer to half than for the baseline problem with 300 weapons. It therefore appears that as the size of the problem increases this fraction tends towards one half. Finally note that if 150 weapons are used in the first stage the lower bound on the value of the solution is 10.54. This again shows that using half of the weapons in the first stage results in a near-optimal solution.

Problem 10: Baseline Problem with higher kill probability but less weapons

The baseline problem except that the kill probability of each weapon-target pair in each of the stages is 0.8 and the defense has 100 weapons:

Static Case:

Optimal solution: [0,0,0,0,0,20,20,20,20,20]

Optimal value: 3.32

Dynamic Case:

Number of weapons used in first stage: 70

Assignment of these weapons: [0,0,0,10,10,10,10,10,10,10]

Lower bound on value of this solution: 6.25

Upper bound on optimal value: 6.53

Remarks:

For this problem we have decreased the number of weapons but increased their kill probability. Note that although there are few weapons the dynamic strategy can still make more effective use of them than the static one. Also note that we can consider the defense as having 80 perfect weapons (Mp). If we look at the baseline problem with a kill probability of 0.5 then the defense can be considered as having 100 weapons. However the optimal value for the former problem is about 6.4 while that of the latter is about 5.7. This indicates that looking at the problem in these terms (i.e.

perfect weapons) can be very misleading. However, since 200 weapons are used for the baseline problem and there are 100 targets let us consider the equivalent kill probability if a target is double shot. Since $p = .5$ then the equivalent kill probability of two weapons is 0.75. This corresponds to 75 perfect weapons. Using this approach we find that the baseline problem with a kill probability of 0.5 should perform worse and indeed it does.

Problem 11: Baseline problem with different asset values

The baseline problem except that the asset values are given by $\vec{W} = [1,1,1,1,1,3,3,3,3,3]$. Note that the maximum possible expected value is 20.

Static Case:

Optimal solution: [0,0,0,0,0,40,40,40,40,40]

Optimal value: 11.57

Dynamic Case:

Number of weapons used in first stage: 110

Assignment of these weapons: [0,0,0,0,10,20,20,20,20,20]

Lower bound on value of this solution: 16.19

Upper bound on optimal value: 17.94

Remarks:

Note that the optimal solution of the static problem is the same as for the baseline problem. Since all of the larger valued assets are defended, the optimal value is three times that for the baseline problem. On the other hand the optimal solution for the dynamic case is to defend all of the larger valued assets with 20 weapons each and to defend one of the unit valued assets with 10 weapons. Recall that in the baseline problem 9 of the assets were defended in the first stage. Note that in this case 81% of the total asset value is saved compared to 71% for the baseline problem. This is expected because, since this problem is non-uniform, the lower valued assets can be left undefended when a preferential defense is used.

Problem 12: Baseline problem with different kill probabilities

The baseline problem except that the kill probabilities in each stage is given by

$\vec{p}(t) = [.5,.5,.5,.5,.5,.68,.68,.68,.68,.68]$.

Static Case:

Optimal solution: [0,0,0,0,0,40,40,40,40,40]

Optimal value: 4.50

Dynamic Case:

Number of weapons used in first stage: 90

Assignment of these weapons: [0,10,10,10,10,10,10,10,10,10]

Lower bound on value of this solution: 6.94

Upper bound on optimal value: 7.23

Remarks:

Note that the kill probabilities were chosen so $(1-.5)(1-.68) = (1-.6)^2$. In other words double shooting in the baseline problem is equivalent in lethality to double shooting in this problem with one low kill probability weapon and one high kill probability weapon. Note that the performance of the static case is better than the performance of the static case for the baseline problem. On the other hand the performance for the dynamic case is roughly the same as that for the baseline problem. Therefore the effect of differing kill probabilities is smaller in the dynamic problem.

Problem 13: Baseline problem with different targets per asset

The baseline problem except that the number of targets aimed at each asset is given by $\vec{n} = [5,5,5,5,5,15,15,15,15,15]$.

Static Case:

Optimal solution: [5,5,5,5,5,0,0,0,0,75]

Optimal value: 5.61

Dynamic Case:

Number of weapons used in first stage: 100

Assignment of these weapons: [5,5,5,5,5,0,0,15,30,30]

Lower bound on value of this solution: 7.65

Upper bound on optimal value: 7.81

Remarks:

The performance for the dynamic case is better than that for the baseline problem. Again this is due to the fact that the number of targets per asset is not the same for all assets but the average number of targets per asset is the same as for the baseline problem. Therefore when a preferential

defense is required the assets with many targets aimed for them will be left undefended while the others would be defended.

7 Concluding Remarks

In this paper we have presented the Dynamic Asset-Based problem together with a sub-optimal algorithm for finding a good solution. We have also presented a method for obtaining an upper bound on the optimal value. In our numerical results we have presented examples which illustrate various properties of the solution of the dynamic problem. We also performed comparisons of the dynamic and static strategies.

This is an extremely difficult problem both analytically and computationally. Because of the difficulty of the problem it is necessary to make approximations. Furthermore, the value of an assignment cannot, in practice, be evaluated exactly because of the number of operations required. Therefore this value must be estimated with the use of simulations.

The sub-optimal algorithm presented, performed well on the problems on which it was run. We believe that if this method is used on the more general version of the problem it will also perform well. In general we have found that the performance of a dynamic strategy is roughly twice that for the corresponding static strategy. An equivalent statement is that half as many weapons are required for the dynamic problem to obtain the same level of performance as the static one. These results show the importance of using a dynamic approach. The increased computational complexity can be reduced by using approximations. We have found that simple approximations reduce the computational complexity while only slightly degrading the performance.

References

[1] P. Hosein and M. Athans, "Preferential Defense Strategies. Part I: The Static Case," this issue.

[2] P. A. Hosein, *A Class of Dynamic Nonlinear Resource Allocation Problems.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA., Sept. 1989.