Naval Research Laboratory

Washington, DC 20375-5000

NRL Memorandum Report 6553

Optimal Resource Allocation for a Controller with Two Resource Types

KIMBERLY M. POTTER

Advanced Techniques Branch
Tactical Electronic Warfare Division

October 11, 1989

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

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 6553	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Naval Research Laboratory 6b. OFFICE SYMBOL (If applicable) Code 5755.1	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 Office of Naval Technology 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code) Arlington, VA 22217	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. 63270N; PROJECT NO. R2030E0T; TASK NO.; WORK UNIT ACCESSION NO. DN156-065

11. TITLE (Include Security Classification)
Optimal Resource Allocation for a Controller with Two Resource Types

12. PERSONAL AUTHOR(S)
Potter, Kimberly, M.

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Interim	FROM 8/87 TO 5/89	1989 October 11	84

16. SUPPLEMENTARY NOTATION

17. COSATI CODES		
FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
Optimal control, Neural networks, EW resource allocation, C3I, Antiship cruise missile defense.

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

A stationary target in a hostile environment is equipped with two types of defense resources. Attackers arrive in a finite time horizon; each with the goal of hitting the target. The success of a defensive resource in countering an attack is probabilistic. The problem of allocating the defensive resources so as to maximize the targets probability of survival is considered. Combinatorial algorithms are given for determining the optimal allocation schemes under deterministic, minimax, and Bayesian formulations, when the arrival times of the attackers are known *a priori*. A neutral network is also applied to find the optimal allocation schemes in the deterministic formulation. For the minimax formulation, an algorithm is given for determining the optimal allocation schemes when the arrived times of the attackers are not known *a priori*.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Sheldon Wolk	22b. TELEPHONE (Include Area Code) (202) 7673178 22c. OFFICE SYMBOL Code 5755.1

DD Form 1473, JUN 86 *Previous editions are obsolete.* SECURITY CLASSIFICATION OF THIS PAGE

S/N 0102-LF-014-6603

Contents

OPTIMAL RESOURCE ALLOCATION FOR A CONTROLLER WITH TWO RESOURCE TYPES

Introduction

In this thesis, the problem of allocating the defensive resources of a target under attack so as to maximize the probability of target survival is considered. A certain number of attackers arrive in a finite time horizon $[0,\Omega]$, each with the goal of hitting the target. The target is equipped with various defensive resources to counter the attacks. Whether or not the target survives depends on the success of the countermeasures. Combinatorial algorithms are developed for determining the optimal allocation schemes under deterministic, minimax, and Bayesian formulations, when the arrival times and arrival angles of the attackers are known a priori, and in the minimax formulation, when they are not. A neural network is developed for determining the optimal allocation scheme under a deterministic formulation when the arrival times and arrival angles of the attackers are known a priori.

Manuscript approved September 9, 1989.

The model used in the analysis makes four assumptions. The first assumption is that the target is stationary. This assumption simplifies the dynamics of the system. As will be seen in Chapter 1, among other variables, the probability of target survival is dependent upon the physical location of the attacker relative to the target. If the target is allowed to move, the probability of target survival then depends on its trajectory. Restricting its motion eliminates that time dependency from the expression for the probability of target survival. Also if the target was allowed to move, its velocity would be much less than the velocity of the attackers, so this assumption is not unreasonable.

The second assumption is that there is only one type of attacker. This assumption simplifies the modeling of the problem, and also simplifies the cost function by removing any dependency on attacker type.

The third assumption is that the only defensive resources available to the target are decoys and a hardkill system. No generality is lost by this assumption.

The fourth assumption is that the target survives only if none of the attackers that arrive in $[0,\Omega]$ hit it. As will be shown in Chapter 1, this assumption is the foundation of the cost function, and results in the determination of the allocation schemes with the highest probability of the target sustaining no hits.

The remainder of this introduction describes the physical model in more detail beginning with the characteristics of the attackers, then an explanation of the operation and application of the countermeasures.

The environment is considered to be a plane, with the center of the target at the origin. All attackers arrive at an initial range of r_A from the target, and at an initial angle in one of M sectors. The sectors are denoted by S_l for

l=1,...,M. The attackers operate by sampling a region of the environment, and then traveling with velocity v_A towards the center of mass of the distribution of returned energy from the objects contained inside the region. This is known as tracking. The sampling region is called the range gate of the attacker. The attackers can only track objects that are inside their range gates.

The range gate width is divided into two segments: the early-gate segment and the late-gate segment. If the attacker first amplifies the energy in the early-gate segment, then centroids on the distribution of energy in the whole gate, it is employing leading-edge tracking (LE). If the attacker centroids on the distribution of energy in the whole gate, treating both segments equally, it is employing centroiding tracking (CEN).

The attackers cannot alternate between tracking techniques, this parameter is fixed for each attacker.

An attacker is either in the tracking state (T), in which it is tracking the target, or in the not-tracking state ($\bar{T}$), in which it is not tracking the target. After a certain amount of time μ_T since a particular attacker's arrival, it will become known to the target which state the attacker is in. Upon arrival and during the period μ_T, it is assumed the attacker is in the tracking state. If an attacker transitions from the tracking state into the not-tracking state, it cannot re-enter the tracking state. If an attacker never transitions out of the tracking state, it will hit the target $\mu_\Omega = \frac{r_A}{v_A}$ time units after its arrival.

Each attacker at time t is characterized by the following parameters:

$$\text{range from the target } r(t) \in \mathcal{R}^+$$

$$\text{angle from the target } \theta(t) \in \{S_1, ..., S_M\}$$

$$\begin{aligned} \text{tracking state } \beta(t) &\in \{T, \bar{T}\} \\ \text{tracking technique } \alpha &\in \{LE, CEN\} \\ \text{arrival time } \tau &\in [0, \Omega]. \end{aligned}$$

The state vector describing the the i^{th} attacker at time t is given by:

$$(r_i(t), \theta_i(t), \beta_i(t), \alpha_i, \tau_i).$$

The target may or may not have knowledge of the tracking technique of the attackers. When the tracking techniques are not known, an a priori probability distribution describing the tracking method may be available. This distribution is given by:

$$\begin{aligned} Pr(\alpha = LE) &= p \\ Pr(\alpha = CEN) &= 1 - p, \end{aligned}$$

and it is assumed the tracking method of one attacker does not depend on the tracking method of any other attacker. Also the target may be provided with periodic updates of the range, angle, and tracking state of each attacker. The arrival times and arrival angles of the attackers may or may not be known a priori. If the arrival schedule is not known in advance, it is assumed the attackers arrive independently of each other according to a Poisson process with stationary arrival rate λ, and with an equal probability of arriving from any one of the M angle sectors.

The target has three countermeasures it can apply to protect itself from the attackers. The first two countermeasures are decoys and the third is a hardkill system. A countermeasure is successful against an attacker if it causes the

attacker to transition from T to $\bar{T}$. Since an attacker cannot transition back to the T state once it has entered the $\bar{T}$ state, a countermeasure can succeed against a particular attacker only if all the countermeasures applied earlier failed.

There are a total of D_L decoys of type L, and D_R decoys of type R available to the target. Deployed decoys are the only objects in the environment for the attackers to track other than the target. Both decoy types operate by entering the range gate of an attacker, then traveling through it with velocity $v_D \ll v_A$ away from the target. The attacker tries to distinguish between the target and the decoy. If the decoy is successful, the attacker will track the decoy's motion instead of tracking the target. The attacker's range gate will move with the decoy, and since the decoy is traveling away from the target, the target will eventually be outside the range gate. The attacker will therefore miss the target. This is known as range gate pull off.

All objects inside the range gate of an attacker influence its decision on which object is the true target, however the attacker must ultimately choose only one object to track. In other words, the attacker's range gate cannot follow more than one decoy. Hence a decoy can succeed in causing an attacker to track it only if all the decoys deployed earlier failed.

Decoys cannot succeed against any attacker whose range is within a radius $r_D \ll r_A$ from the target. Every time the target deploys either type of decoy, each one is placed in the same physical location with respect to the target, with the location for the type L decoys different from that for the type R decoys. It is assumed that when a decoy is deployed, it will be contained inside the range gate of any attacker that is presently tracking the target with a range greater

than r_D. The decoy is also contained inside the range gate of any attacker that arrives within μ_D time units after the deployment.

There must be a delay of Δ_D time units between deploying decoys of the either type. This is to prevent the decoys from stacking on top of each other, as stacked decoys are not better at pulling the range gate of an attacker off the target than single decoys.

The third countermeasure is a hardkill system. There are a total of B units of this resource available to the target. One unit is used each time the system is employed. The target operates this system by directing it toward a specific attacker and firing. If the system is successful, the attacker will immediately stop tracking the target. The target is then alerted of the change in the attacker's state. The system has a maximum range of $r_B \leq r_D$, outside of which it is totally ineffective. The target must wait Δ_B time units after using the hardkill system before using it again. This countermeasure is denoted by G_i, where the subscript i signifies which attacker the hardkill system is directed toward.

When the attackers arrive, and until it is known otherwise, they are assumed to travel radial trajectories toward the target given by:

$$\begin{aligned} r(t) &= r_A - v_A(t-\tau);\ \tau - \mu_D \leq t \leq \tau + \mu_\Omega \\ \theta(t) &= \theta(\tau);\ \tau - \mu_D \leq t \leq \tau + \mu_\Omega. \end{aligned}$$

An attacker will only veer from this trajectory if a countermeasure has caused it to enter the not-tracking state.

The time interval $[0,\Omega]$ is broken into decision instants. The decision instants are the only times at which the target can apply countermeasures. If updates of the states of the attackers are available, they will be provided at each decision

instant. There is a decision instant every Δ time units starting at time 0, resulting in a total of $K = \lfloor \frac{\Omega}{\Delta} \rfloor$ decision instants. The times Δ_D and Δ_B are both multiples of Δ. The k^{th} decision instant occurs at time t_k. A decoy of type L or of type R is a feasible countermeasure at the k^{th} decision instant if t_k does not occur during the period Δ_D after the most recent deployment of a decoy, and there is either at least one attacker present with range greater than r_D, or at least one attacker known to have an arrival time in the interval $[t_k, t_k + \mu_D]$. Also the supply of decoys must not be exhausted. The hardkill system is a feasible countermeasure at the k^{th} decision instant if t_k does not occur during the period Δ_B after the last use of the system, and there is at least one attacker with range less than r_B. Also the supply of resource used by the hardkill system must not be exhausted.

Let $\mathcal{U}_k$ denote the set of feasible countermeasures or controls at the k^{th} decision instant. All sets include a "do nothing" control, which is denoted by Z. At time t_k the target chooses to apply one and only one countermeasure from $\mathcal{U}_k$.

Let u_k denote the countermeasure or control to apply at the k^{th} decision instant, where $u_k \in \mathcal{U}_k$, and let $\vec{u}_K = (u_1, u_2, ..., u_K)$ be the allocation scheme, or control vector, for K decision instants.

It is desired to determine the allocation scheme, or the control vector, that maximizes the target's probability of survival. In this thesis, combinatorial algorithms are developed for determining the optimal schemes under various conditions. When the arrival times and arrival angles of the attackers are known a priori, an exhaustive search and a trimmed search are developed for determining

the optimal allocation schemes under a deterministic formulation, in which the tracking methods of the attackers are known a priori; under a minimax formulation in which the tracking methods are never known to the target, and there is no prior probability distribution available describing this parameter; and under a Bayesian formulation, in which the tracking methods are again unknown, but there is a prior probability distribution available describing the tracking method. In the minimax formulation, the trimmed search is applied to determine the optimal allocation schemes when the arrival times are not known a priori. A neural network of the type applied to the Traveling Salesman Problem by Hopfield and Tank [1], is developed for obtaining the optimal allocation scheme in the deterministic formulation, when the arrival times and arrival angles of each attacker are known a priori. In Chapter 1 the probability of target survival is derived. In the remaining chapters, the algorithms are described. The efficiency with which each algorithm arrives at a solution, and the ability of each algorithm to determine the optimal control sequence are discussed. Numerical results from applying the algorithms to specific attack situations are given in Chapter 5.

Chapter 1

Derivation of Cost Function

As stated in the introduction, the objective of each algorithm is to determine the sequence of countermeasures that maximizes the probability of target survival. From the fourth model assumption, the target survives only if none of the attackers arriving in $[0,\Omega]$ hit it. Whether or not an attacker hits the target depends on the success of the countermeasures applied by the target. For each countermeasure, there is a probability distribution describing its success in causing an attacker to transition from T to $\bar{T}$, as a function of the time the countermeasure is applied, and the state vector of the attacker at the time of the application.

Since a countermeasure can succeed in causing an attacker to transition from the tracking state into the not-tracking state only if all the countermeasures applied earlier failed, the probability of success of a countermeasure against an attacker is zero if the attacker is in the not-tracking state at the time the countermeasure is applied. It is also zero if the countermeasure is applied after time $\tau + \mu_\Omega$, where τ is the attacker's arrival time.

The probability of success of a type L decoy deployed at time t_k against the i^{th} attacker, assuming all countermeasures applied previous to t_k failed, is given by

$$P_L(r_i(t_k), \theta_i(t_k), \beta_i(t_k), \alpha_i, \tau_i, t_k).$$

Similarly, the probability of success of a type R decoy deployed at time t_k against the i^{th} attacker, assuming all countermeasures applied previous to t_k failed, is given by

$$P_R(r_i(t_k), \theta_i(t_k), \beta_i(t_k), \alpha_i, \tau_i, t_k).$$

Unlike the decoys, the target must direct the hardkill system toward a specific attacker. The probability of success of the system is nonzero only for the attacker it is aimed at; it is zero for all other attackers. The probability of success of the hardkill system directed toward the i^{th} attacker at time t_k, assuming all countermeasures applied previous to t_k failed, is given by

$$P_G(r_i(t_k), \theta_i(t_k), \tau_i, t_k).$$

All three of the above distributions are known.

Since the probability distribution describing the success of the hardkill system does not depend on the attacker state parameters α and $\beta(t)$, they can

be included as parameters of the distribution without changing its shape. This gives the probability of success of the hardkill system directed toward the i^{th} attacker at time t_k, assuming all countermeasures applied previous to t_k failed, as

$$P_G(r_i(t_k), \theta_i(t_k), \beta_i(t_k), \alpha_i, \tau_i, t_k).$$

Since the three distributions describing the success of the countermeasures all depend on the same variables, one distribution can be defined that describes the probability of success of a countermeasure as a function of the applied countermeasure, the time of the application, and the state vector of the attacker at the time of the application.

Define the following two events:

S_j^i : Countermeasure applied at time t_j caused the i^{th} attacker to transition from T to $\bar{T}$

F_j^i : Countermeasure applied at time t_j failed to cause the i^{th} attacker to transition from T to $\bar{T}$.

The probability of success of a countermeasure u_k against the i^{th} attacker at time t_k, assuming all countermeasures applied previous to t_k failed, is then given by

$$Pr(\mathrm{S}_k^i|\mathrm{F}_{k-1}^i \ldots \mathrm{F}_1^i) \;=\; P(r_i(t_k), \theta_i(t_k), \beta_i(t_k), \alpha_i, \tau_i, t_k, u_k). \qquad (1.1)$$

For notational convenience, the above distribution will be represented by

$$Pr(\mathrm{S}_k^i|\mathrm{F}_{k-1}^i \ldots \mathrm{F}_1^i) \;=\; P_{u_k}^i(t_k).$$

1.1 Known Arrival Times

Suppose the arrival times and the arrival angles of the attackers are known to the target a priori. This means the total number of attackers will be known. Let n be the number of attackers arriving in $[0,\Omega]$. Define the following two events:

$$M_i : \text{The } i^{th} \text{ attacker misses the target}$$

$$H_i : \text{The } i^{th} \text{ attacker hits the target.}$$

The probability of target survival is written as

$$\begin{aligned} &Pr(\text{Target survival}) \\ &\quad = Pr(\text{None of the } n \text{ attackers hit the target}) \\ &\quad = Pr(\text{All } n \text{ attackers miss the target}) \\ &\quad = Pr(\text{Miss}) \\ &\quad = Pr(M_1 \cap M_2 \cap ... \cap M_n). \end{aligned} \tag{1.2}$$

The success or failure of an attacker in hitting the target may affect the chances of the other attackers. Therefore, it cannot be assumed that the events $M_1, M_2, ..., M_n$ are independent. Hence,

$$\begin{aligned} Pr(\text{Miss}) &= Pr(M_1 \cap M_2 \cap ... \cap M_n) \\ &= Pr(M_n|M_{n-1}...M_1)Pr(M_{n-1}|M_{n-2}...M_1)...Pr(M_1) \\ &= \prod_{i=1}^{n} Pr(M_i|M_{i-1}...M_1) \end{aligned}$$

$$= \prod_{i=1}^{n}[1 - Pr(\mathrm{H}_i|\mathrm{M}_{i-1}...\mathrm{M}_1)]. \quad (1.3)$$

Now a hit by an attacker occurs if the attacker remains in the tracking state during the period $[\tau, \tau + \mu_\Omega]$, where τ is the attacker's arrival time. There are two ways in which this can happen. The first way is if no counter-action is taken by the target against the attacker during $[\tau, \tau + \mu_\Omega]$. Recall that an attacker is tracking the target upon arrival. If the target takes no defensive measures, the attacker will continue tracking it.

The second way an attacker will remain in the tracking state is if all the counter-actions taken by the target during the time $[\tau, \tau + \mu_\Omega]$ fail to cause the attacker to transition into the not-tracking state.

The probability of a hit by the i^{th} attacker, assuming the i-1 attackers arriving before τ_i each missed the target, is given by

$$\begin{aligned} Pr(\mathrm{H}_i|\mathrm{M}_{i-1}...\mathrm{M}_1) &= Pr(F_1^i \cap F_2^i \cap ... \cap F_K^i) \\ &= Pr(F_K^i|F_{K-1}^i...F_1^i)...Pr(F_1^i) \\ &= \prod_{k=1}^{K}[1 - Pr(S_k^i|F_{k-1}^i...F_1^i)] \\ &= \prod_{k=1}^{K}[1 - P_{u_k}^i(t_k)], \end{aligned} \quad (1.4)$$

where K is the total number of decision instants in $[0,\Omega]$.

The probability of target survival given in (1.3) then becomes

$$\begin{aligned} Pr(\mathrm{Miss}) &= \prod_{i=1}^{n}[1 - Pr(\mathrm{H}_i|\mathrm{M}_{i-1}...M_1)] \\ &= \prod_{i=1}^{n}(1 - \prod_{k=1}^{K}[1 - P_{u_k}^i(t_k)]). \end{aligned} \quad (1.5)$$

The above expression for the probability of target survival can be written in a second form. An attacker will miss the target if one of the countermeasure

applied in $[\tau, \tau + \mu_\Omega]$ causes the attacker to transition into the not-tracking state. The probability of a miss by the i^{th} attacker, assuming the i-1 attackers arriving before t_i each missed the target, is therefore given by

$$\begin{aligned} Pr(\mathrm{M_i}|\mathrm{M_{i-1}}...\mathrm{M_1}) &= Pr(S_K^i F_{K-1}^i ... F_1^i \cup ... \cup S_2^i F_1^i \cup S_1^i) \\ &= Pr(S_K^i F_{K-1}^i ... F_1^i) + ... + Pr(S_2^i F_1^i) + Pr(S_1^i) \\ &= [P_{u_k}^i(t_k)(1 - P_{u_{k-1}}^i(t_k))...(1 - P_{u_1}^i(t_k))] + ... \\ &\quad + [P_{u_2}^i(t_k)(1 - P_{u_1}^i(t_k))] + P_{u_1}^i(t_k) \\ &= \sum_{k=1}^{K} P_{u_k}^i(t_k) \prod_{l=1}^{k-1}(1 - P_{u_l}^i(t_k)) \\ &= \sum_{k=1}^{K} \prod_{l=1}^{k-1} P_{u_k}^i(t_k)(1 - P_{u_l}^i(t_k)). \end{aligned} \tag{1.6}$$

Using (1.6), the probability of target survival in (1.3) can also be expressed as

$$\begin{aligned} Pr(\mathrm{Miss}) &= \prod_{i=1}^{n} Pr(\mathrm{M_i}|\mathrm{M_{i-1}}...M_1) \\ &= \prod_{i=1}^{n} [\sum_{k=1}^{K} \prod_{l=1}^{k-1} P_{u_k}^i(t_k)(1 - P_{u_l}^i(t_k))]. \end{aligned} \tag{1.7}$$

Since the events $\mathrm{M_i}$ and $\mathrm{H_i}$ exhaust the sample space, (1.4) and (1.6) can be combined to yield the following relationship:

$$\sum_{k=1}^{K} \prod_{l=1}^{k-1} P_{u_k}^i(t_k)(1 - P_{u_l}^i(t_k)) + \prod_{k=1}^{K} [1 - P_{u_k}^i(t_k)] = 1. \tag{1.8}$$

1.2 Unknown Arrival Times

Now suppose neither the arrival times nor the arrival angles of the attackers are known to the target a priori. The number of attackers arriving in [0,Ω] is

therefore not known. The attackers arrive according to a Poisson process with stationary arrival rate λ. The expected number of attackers, n, arriving in [0,Ω] is $\lambda\Omega$. The probability that exactly m attackers arrive in [0,Ω] is given by

$$Pr(\mathrm{n} = m) = \frac{(\lambda\Omega)^m e^{-\lambda\Omega}}{m!}. \tag{1.9}$$

Using the principle of total probability, the probability of target survival becomes

$$Pr(\mathrm{Miss}) = \sum_{m=0}^{\infty} Pr(\mathrm{Miss}|\mathrm{n} = m)Pr(\mathrm{n} = m).$$

The quantity $Pr(\mathrm{Miss}|\mathrm{n} = m)$ is given by (1.3) with n=m, that is

$$Pr(\mathrm{Miss}|\mathrm{n} = m) = \prod_{i=1}^{m}[1 - Pr(\mathrm{H_i}|\mathrm{M_{i-1}}...\mathrm{M_1})].$$

Hence, the probability of target survival can be written as

$$\begin{aligned} Pr(\mathrm{Miss}) &= \sum_{m=0}^{\infty} Pr(\mathrm{Miss}|\mathrm{n} = m)Pr(\mathrm{n} = m) \\ &= \sum_{m=0}^{\infty} \frac{(\lambda\Omega)^m e^{-\lambda\Omega}}{m!} \prod_{i=1}^{m}[1 - Pr(\mathrm{H_i}|\mathrm{M_{i-1}}...\mathrm{M_1})] \\ &= \sum_{m=0}^{\infty} \frac{(\lambda\Omega)^m e^{-\lambda\Omega}}{m!} \prod_{i=1}^{m}[1 - \prod_{k=1}^{K}(1 - P^i_{u_k}(t_k))]. \end{aligned} \tag{1.10}$$

Using the relationship in Equation (8), this can also be written as

$$Pr(\mathrm{Miss}) = \sum_{m=0}^{\infty} \frac{(\lambda\Omega)^m e^{-\lambda\Omega}}{m!} \prod_{i=1}^{m}[\sum_{k=1}^{K} \prod_{l=1}^{k-1} P^i_{u_k}(t_k)(1 - P^i_{u_l}(t_k))]. \tag{1.11}$$

Two expressions of the cost function as the probability of target survival have now been derived for both the case of known attacker arrival times and arrival angles, and the case of unknown attacker arrival times. The form used depends on the attacker arrival information available to the target, and on which algorithm is being applied to find the optimal sequence of countermeasures.

Chapter 2

Known Arrival Times

Suppose the attacker arrival times and arrival angles are known to the target a priori. Let n be the total number of attackers arriving in $[0,\Omega]$ such that $0 < \tau_1 < ... < \tau_n < \Omega$.

The time interval $[0,\Omega]$ can be defined in terms of the arrival times of the first and n^{th} attackers. Recall from the introduction, that a decoy deployed at time t can affect the tracking state of any attacker whose range from the target at time t is greater than r_D, including any attacker that arrives within μ_D time units after the deployment. Therefore time 0 can be defined as $\tau_1 - \mu_D$. The target has μ_Ω time units after an attacker's arrival in which to counter its

attack. The time Ω can therefore be defined as $\tau_n + \mu_\Omega$. The time interval $[0,\Omega]$ is then given by $[\tau_1 - \mu_D, \tau_n + \mu_\Omega]$, and the total number of decision instants is $K = \lfloor \frac{\tau_n - \tau_1 + \mu_D + \mu_\Omega}{\Delta} \rfloor$.

As defined earlier, $\vec{u}_K = (u_1, ..., u_K)$ is the allocation scheme, or control vector, for K decision instants, where $u_k \in \mathcal{U}_k$ is the countermeasure, or control, to apply at the k^{th} decision instant, and $\mathcal{U}_k$ is the set of countermeasures, or controls, feasible at time t_k.

2.1 States Updates Unavailable

In each of the three formulations, in order to determine the optimal allocation scheme for a given set of arrival times and arrival angles, a set of feasible allocation schemes, or feasible control vectors, must be defined. It is necessary to know the states of the attackers at each decision instant, and the previously applied controls, in order to determine the set of controls feasible at each decision instant, and hence the set of feasible control vectors. However, since no updates on the attackers states will be provided, the target must predict which controls will be feasible at each decision instant. This is done by predicting the states of the attackers at each decision instant. A set of feasible control vectors based on these predictions is then determined. The target assumes each attacker maintains a radial trajectory toward the target given by

$$\begin{aligned} r_i(t) &= r_A - v_A(t - \tau_i) \; ; \; \tau_i - \mu_D \leq t < \tau_i + \mu_\Omega \\ \theta_i(t) &= \theta_i(\tau_i) \qquad\quad ; \; \tau_i - \mu_D \leq t < \tau_i + \mu_\Omega \end{aligned}$$

$$\beta_i(t) = \begin{cases} T & \tau_i - \mu_D \leq t < \tau_i + \mu_\Omega \\ \overline{T} & \text{otherwise,} \end{cases} \tag{2.1}$$

for the i^{th} attacker.

These states are used to predict the time periods during which each control will be feasible. Recall that a decoy deployed at time t cannot affect the tracking state of any attacker whose range is within a radius r_D of the target at time t. The i^{th} attacker will reach the range r_D at time $\tau_i + \frac{r_A - r_D}{v_A}$, assuming it maintains a radial trajectory toward the target. Either decoy is therefore a feasible control every Δ_D time units during the periods $[\tau_i - \mu_D, \tau_i + \frac{r_A - r_D}{v_A}]$ for i=1,...,n. Recall also that an attacker must be within a range r_B of the target in order to employ the hardkill system against that attacker. The i^{th} attacker will reach the range r_B at time $\tau_i + \frac{r_A - r_B}{v_A}$, assuming it maintains a radial trajectory toward the target. The target can fire on the i^{th} attacker every Δ_B time units until time $\tau_i + \mu_\Omega$. The hardkill system is therefore feasible control against the i^{th} attacker during the time period $[\tau_i + \frac{r_A - r_B}{v_A}, \tau_i + \mu_\Omega]$. If a decision instant occurs at a time when neither a decoy nor the hardkill system is feasible, the only control available to the target is the Z (do nothing) control.

A set of predicted feasible control vectors can then be determined combinatorially from knowledge of the controls predicted to be feasible at each decision instant. The set $\mathcal{U}_k$ contains the controls predicted to be feasible at time t_k. The number of predicted feasible control vectors is given by:

$$\prod_{k=1}^{K} | \mathcal{U}_k |,$$

where $| \mathcal{U}_k |$ is the cardinality of the set $\mathcal{U}_k$.

Consider a tree with K levels, where each level corresponds to a decision instant. At the k^{th} level of the tree, there are $\prod_{i=1}^{k-1} \mid \mathcal{U}_i \mid$ nodes with $\mid \mathcal{U}_k \mid$ branches emanating from each. The branches represent the feasible controls at time t_k. Every complete path through the tree represents a feasible control vector.

In each of the three formulations, the same set of predicted feasible control vectors is searched to find the respective optimal allocation schemes, that is there is only one set of predicted feasible control vectors.

2.1.1 Deterministic Formulation

Suppose the tracking methods of the attackers are known to the target a priori. The arrival state of the i^{th} attacker is given by

$$(r_i(\tau_i) = r_A, \theta_i(\tau_i), \beta_i(\tau_i) = T, \alpha_i, \tau_i),$$

where $\theta_i(\tau_i)$, τ_i, and α_i are given.

Using the expression for the cost function given in (1.7), the optimal allocation scheme is the control vector which achieves the following:

$$\max_{\vec{u}_K}\{\prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1} P^i_{u_k}(t_k)(1 - P^i_{u_l}(t_l))]\}. \tag{2.2}$$

The solution vector is denoted by $\vec{u}_{opt}$.

Computing $\vec{u}_{opt}$ requires computing (1.7) for every path in the tree, using the state predicting equations in (2.1), and then comparing the resulting $\prod_{k=1}^{K} \mid \mathcal{U}_k \mid$ quantities to find the maximum.

Although this exhaustive search is guaranteed to find the solution to (2.2), this method is undesirable if K is large. Instead of exhaustively searching the

set of predicted feasible control vectors, the same $\vec{u}_{opt}$ can be obtained by exhaustively searching only the set of controls predicted to be feasible at *each decision instant*, determining $\vec{u}_{opt}$ element by element.

At a given level in the tree, all branches leaving a specific node are compared. One branch is selected which leads to one of the nodes at the next level. The control associated with the surviving branch is the control applied at the decision instant corresponding to that level. In this manner, the optimal path is traced out sequentially. Consider the following algorithm.

Algorithm 1:

(1) At time t_1, apply the control that achieves

$$\max_{u_1}\{\prod_{i=1}^{n} P^i_{u_1}(t_1)\}.$$

Denote this control by u_1^*.

(2) At time t_2, apply the control that achieves

$$\max_{u_2}\{\prod_{i=1}^{n}[P^i_{u_1^*}(t_1) + P^i_{u_2}(t_2)(1 - P^i_{u_1^*}(t_1))]\}.$$

Denote this control by u_2^*.

(3) At time t_m, apply the control that achieves

$$\max_{u_m}\{\prod_{i=1}^{n}[[\sum_{k=1}^{m-1}\prod_{l=1}^{k-1} P^i_{u_k^*}(t_k)(1 - P^i_{u_l^*}(t_l))] + P^i_{u_m}(t_m)\prod_{l=1}^{m-1}(1 - P^i_{u_l^*}(t_l))]\}.$$

Denote this control by u_m^*.

(4) Continue until K controls have been determined. The optimal control vector is given by $\vec{u}^{A1}_{opt} = (u_1^*, ..., u_K^*)$.

The cost of $\vec{u}^{A1}_{opt}$ is given by:

$$J(\vec{u}^{A1}_{opt}) = \prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1} P^i_{u_k^*}(t_k)(1 - P^i_{u_l^*}(t_l))]. \qquad (2.3)$$

Example:

Suppose n=1 (one missile), K=2, and there are only two countermeasures: 0 and 1. Suppose also the probability of success of the countermeasures are time invariant. Let $P_0(t) = a$ and $P_1(t) = b$, where $a > b$.

Exhaustive Search:

$$\begin{aligned}
&\max_{\vec{u}_2}\{P_{u_1}(t_1) + P_{u_2}(t_2)(1 - P_{u_1}(t_1))\} \\
&\quad = \max\{\mathrm{a} + \mathrm{a}(1 - \mathrm{a}), \mathrm{a} + \mathrm{b}(1 - \mathrm{a}), \mathrm{b} + \mathrm{b}(1 - \mathrm{b})\} \\
&\quad = \mathrm{a} + \mathrm{a}(1 - \mathrm{a}) \Rightarrow \vec{u}_{opt} = (0, 0).
\end{aligned}$$

Algorithm 1:

Step 1:

$$\max_{u_1 \in \mathcal{U}_1}\{P_{u_1}(t_1)\} = \max\{\mathrm{a}, \mathrm{b}\} = \mathrm{a} \Rightarrow u_1^* = 0.$$

Step 2:

$$\begin{aligned}
&\max_{u_2 \in \mathcal{U}_2}\{P_{u_1^*}(t_1) + P_{u_2}(t_2)(1 - P_{u_1^*}(t_1))\} \\
&\quad = \max\{\mathrm{a} + \mathrm{a}(1 - \mathrm{a}), \mathrm{a} + \mathrm{b}(1 - \mathrm{a})\} \\
&\quad = \mathrm{a} + \mathrm{a}(1 - \mathrm{a}) \Rightarrow u_2^* = 0.
\end{aligned}$$

Therefore $\vec{u}_{opt}^{A1}$=(0,0)=$\vec{u}_{opt}$.

Proof:

Induction is used to show that Algorithm 1 and the exhaustive search both arrive at the same $\vec{u}_{opt}$. It is desired to show that $J(\vec{u}_{opt}) = J(\vec{u}_{opt}^{A1})$, that is to

show the following:

$$\max_{\vec{u}_K}\{\prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1}[P^i_{u_k}(t_k)(1-P^i_{u_k}(t_l))]\}$$

$$= \prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1}P^i_{u^*_k}(t_k)(1-P^i_{u^*_l}(t_l))]$$

$$= \max_{u_K}\{\prod_{i=1}^{n}[[\sum_{k=1}^{K-1}\prod_{l=1}^{k-1}P^i_{u^*_k}(t_k)(1-P^i_{u^*_l}(t_l))]$$

$$+P^i_{u_K}(t_K)\prod_{l=1}^{K-1}(1-P^i_{u^*_l}(t_l))]\}, \qquad (2.4)$$

where the $u^*_1, ..., u^*_K$ are the K controls determined by Algorithm 1.

Without loss of generality, assume n=1. For notational convenience in this proof, the quantity $P^1_{u_k}(t_k)$ will be denoted by P_{u_k}. Recall $\vec{u}_K = (u_1, ..., u_K)$.

For K=1, the vector $\vec{u}_1$=u_1, and the following is true:

$$\max_{\vec{u}_1}\{P_{u_1}\} = \max_{u_1}\{P_{u_1}\} = P_{u^*_1},$$

so the equality in (2.4) holds.

Now assume (2.4) is true for $K = m$ so that

$$\max_{\vec{u}_m}\{P_{u_1} + P_{u_2}(1-P_{u_1}) + ... + P_{u_m}(1-P_{u_{m-1}})...(1-P_{u_1})\}$$

$$= P_{u^*_1} + ... + P_{u^*_m}(1-P_{u^*_{m-1}})...(1-P_{u^*_1})$$

$$= \max_{u_m \in \mathcal{U}_m}\{P_{u^*_1} + ... + P_{u_m}(1-P_{u^*_{m-1}})...(1-P_{u^*_1})\}. \qquad (2.5)$$

It must now be shown that (2.4) is true for $K = m + 1$, that is

$$\max_{\vec{u}_{m+1}}\{P_{u_1} + ... + P_{u_m}(1-P_{u_{m-1}})...(1-P_{u_1})$$

$$+P_{u_{m+1}}(1-P_{u_m})...(1-P_{u_1})\}$$

$$= P_{u^*_1} + ... + P_{u^*_m}(1-P_{u^*_{m-1}})...(1-P_{u^*_1})$$

$$+P_{u^*_{m+1}}(1-P_{u^*_m})...(1-P_{u^*_1}).$$

$$= \max_{u_{m+1}}\{P_{u^*_1} + ... + P_{u^*_m}(1-P_{u^*_{m-1}})...(1-P_{u^*_1})$$

$$+P_{u_{m+1}}(1-P_{u^*_m})...(1-P_{u^*_1})\} \tag{2.6}$$

Using (2.5) and (1.8), the right hand side of (2.6) becomes

$$= \max_{u_{m+1}}\{\max_{\vec{u}_m}\{P_{u_1} + ... + P_{u_m}(1-P_{u_{m-1}})...(1-P_{u_1})\}$$

$$+P_{u_{m+1}}[1-(P_{u^*_1} + ... + P_{u^*_m}(1-P_{u^*_{m-1}})...(1-P_{u^*_1}))]\}.$$

Using (2.5) again this becomes

$$= \max_{u_{m+1}}\{\max_{\vec{u}_m}\{P_{u_1} + ... + P_{u_m}(1-P_{u_{m-1}})...(1-P_{u_1})\}$$

$$+P_{u_{m+1}}[1-\max_{\vec{u}_m}\{P_{u_1} + ... + P_{u_m}(1-P_{u_{m-1}})...(1-P_{u_1})\}]\}.$$

Grouping terms gives

$$= \max_{u_{m+1}}\{\max_{\vec{u}_m}\{P_{u_1} + ... + P_{u_m}(1-P_{u_{m-1}})...(1-P_{u_1})\}$$

$$\times(1-P_{u_{m+1}}) + P_{u_{m+1}}\}.$$

Since u_{m+1} is not included in a search over $\vec{u}_m$ this becomes

$$= \max_{u_{m+1}}\{\max_{\vec{u}_m}\{[P_{u_1} + ... + P_{u_m}(1-P_{u_{m-1}})...(1-P_{u_1})]$$

$$\times(1-P_{u_{m+1}}) + P_{u_{m+1}}\}\}.$$

Using (1.8) yields the left hand side of (2.6):

$$= \max_{\vec{u}_{m+1}}\{P_{u_1} + ... + P_{u_m}(1-P_{u_{m-1}})...(1-P_{u_1})$$

$$+P_{m+1}(1-P_{u_m})...(1-P_{u_1})\}_{Q.E.D.}$$

Algorithm 1 is a trimmed search. At the k^{th} decision instant, it selects one of the $\mid \mathcal{U}_k \mid$ feasible controls, which means it compares $\mid \mathcal{U}_k \mid$ control vectors

that only differ in the last element, so that from the first level to the K^{th} level, a total of $\sum_{k=1}^{K} \mid \mathcal{U}_k \mid$ path comparisons were made to obtain $\vec{u}_{opt}$, as opposed to a total of $\prod_{k=1}^{K} \mid \mathcal{U}_k \mid$ path comparisons required in an exhaustive search.

2.1.2 Minimax Formulation

Now suppose that the tracking methods of the attackers will never be known to the target, and that no a priori probability distribution describing this parameter is available. Recall that an attacker employs one of two possible tracking methods to decide in which direction to travel: leading-edge tracking (LE) or centroiding tracking (CEN). The tracking method is one of the parameters that determines the probability of a countermeasure successfully causing the attacker to transition from the tracking state into the not-tracking state, and is therefore one of the parameters that determines the probability of target survival. A minimax formulation is applied to find the best allocation scheme in this situation.

Recall from the introduction that α_i denotes the tracking method of the i^{th} attacker, where $\alpha_i \in \{LE, CEN\}$. Let $\vec{\alpha}=(\alpha_1, ..., \alpha_n)$ represent the tracking method vector for n attackers. There are 2^n possible tracking method vectors.

Define the conditional cost $R_m(\vec{u}_K)$ to be the probability of target survival using allocation scheme $\vec{u}_K$, assuming the m^{th} tracking method vector is the true tracking method vector. The target seeks a control vector which maximizes, over all feasible control vectors $\vec{u}_K$, the minimum of the conditional costs:

$$\min\{R_1(\vec{u}_K), R_2(\vec{u}_K), ..., R_{2^n}(\vec{u}_K)\}. \tag{2.7}$$

The control vector achieving this maximum is denoted $\vec{u}_{opt}$.

The arrival state of the i^{th} attacker is given by

$$(\tau_i(\tau_i) = r_A, \theta_i(\tau_i), \beta_i(\tau_i) = T, \alpha_i, \tau_i),$$

where $\theta_i(\tau_i)$ and τ_i are given, and this time α_i is unknown.

Using the expression for the cost function given in (1.7), the cost of $\vec{u}_{opt}$ is written as

$$\begin{aligned} J(\vec{u}_{opt}) &= \max_{\vec{u}_K} \min_{\vec{\alpha}} Pr(\text{Miss}). \\ &= \max_{\vec{u}_K} \min_{\vec{\alpha}} \{\prod_{i=1}^{n} [\sum_{k=1}^{K} \prod_{l=1}^{k-1} P^i_{u_k}(t_k)(1 - P^i_{u_l}(t_l))]\}. \end{aligned} \tag{2.8}$$

Determining $\vec{u}_{opt}$ by exhaustive search requires solving (2.8), which involves calculating the minimum of the conditional costs for every path in the tree, and then comparing the resulting $\prod_{k=1}^{K} \mid \mathcal{U}_k \mid$ quantities to find the maximum.

Instead of exhaustively searching the set of feasible control vectors to find $\vec{u}_{opt}$, a sequential algorithm can be applied that will result in the same $\vec{u}_{opt}$. This algorithm is similar to Algorithm 1, and differs from it only in the criterion by which it selects the survivor path.

Algorithm 2:

(1) At time t_1, apply the control that achieves

$$\max_{u_1} \min_{\vec{\alpha}} \{\prod_{i=1}^{n} P^i_{u_1}(t_1)\}.$$

Denote this control by u_1^*.

(2) At time t_2, apply the control that achieves

$$\max_{u_2} \min_{\vec{\alpha}} \{\prod_{i=1}^{n} [P^i_{u_1^*}(t_1) + P^i_{u_2}(t_2)(1 - P^i_{u_1^*}(t_1))]\}.$$

Denote this control by u_2^*.

(3) At time t_m, apply the control that achieves

$$\max_{u_m} \min_{\vec{\alpha}} \{\prod_{i=1}^{n} [[\sum_{k=1}^{m-1} \prod_{l=1}^{k-1} P^i_{u_k^*}(t_k)(1 - P^i_{u_l^*}(t_l))] + P^i_{u_m}(t_m) \prod_{l=1}^{m-1} (1 - P^i_{u_l^*}(t_l))]\}.$$

Denote this control by u_m^*.

(4) Continue until K controls have been determined. The optimal control vector is given by $\vec{u}^{A2}_{opt} = (u_1^*, ..., u_K^*)$.

The cost of $\vec{u}^{A2}_{opt}$ is given by:

$$J(\vec{u}^{A2}_{opt}) = \prod_{i=1}^{n} [\sum_{k=1}^{K} \prod_{l=1}^{k-1} P^i_{u_k^*}(t_k)(1 - P^i_{u_l^*}(t_l))]. \tag{2.9}$$

Example:

Suppose n=1 (one missile), K=2, and there are only two countermeasures: 0 and 1. Suppose also the probability of success of the countermeasures are time invariant and depend only on the tracking method. For the probability of success of the countermeasures given in the table below, assume b<c<d<a:

α	0	1
LE	a	b
CEN	c	d

Exhaustive Search:

$$\max_{\vec{u}_2} \min_{\vec{\alpha}} \{P_{u_1}(t_1) + P_{u_2}(t_2)(1 - P_{u_1}(t_1))\}$$

$$= \max\{\min[a + a(1 - a), c + c(1 - c)], \min[a + b(1 - a), c + d(1 - c)],$$

$$\min[b + a(1 - b), d + c(1 - d)], \min[b + b(1 - b), d + d(1 - d)]\}$$

$$= c + d(1 - c) \Rightarrow \vec{u}_{opt} = (0, 1).$$

Algorithm 2:

Step 1:

$$\max_{u_1 \in \mathcal{U}_1} \min_{\vec{\sigma}} \{P_{u_1}(t_1)\} = \max\{\min[a, c], \min[b, d]\} = c \Rightarrow u_1^* = 0.$$

Step 2:

$$\max_{u_2 \in \mathcal{U}_2} \min_{\vec{\sigma}} \{P_{u_1^*}(t_1) + P_{u_2}(t_2)(1 - P_{u_1^*}(t_1))\}$$
$$= \max\{\min[a + a(1-a), c + c(1-c)], \min[a + b(1-a), c + d(1-c)]\}$$
$$= c + d(1-c) \Rightarrow u_2^* = 1.$$

Therefore $\vec{u}_{opt}^{A2}=(0,1)=\vec{u}_{opt}$.

Proof:

Induction is again used to show that Algorithm 2 and the exhaustive search both arrive at the same $\vec{u}_{opt}$. It is desired to show that $J(\vec{u}_{opt}) = J(\vec{u}_{opt}^{A2})$, that is to show the following:

$$\max_{\vec{u}_K} \min_{\vec{\sigma}} \{\prod_{i=1}^{n} [\sum_{k=1}^{K} \prod_{l=1}^{k-1} P_{u_k}^i(t_k)(1 - P_{u_k}^i(t_l))]\}$$
$$= \prod_{i=1}^{n} [\sum_{k=1}^{K} \prod_{l=1}^{k-1} P_{u_k^*}^i(t_k)(1 - P_{u_l^*}^i(t_l))]$$
$$= \max_{u_K \in \mathcal{U}_K} \min_{\vec{\sigma}} \{\prod_{i=1}^{n} [[\sum_{k=1}^{K-1} \prod_{l=1}^{k-1} P_{u_k^*}^i(t_k)(1 - P_{u_l^*}^i(t_l))]$$
$$+ P_{u_K}^i(t_K) \prod_{l=1}^{K-1} (1 - P_{u_l^*}^i(t_l))]\}. \qquad (2.10)$$

where the $u_1^*, ..., u_K^*$ are the K controls determined by Algorithm 2.

Without loss of generality, assume n=1. For notational convenience in this proof, the quantity $P_{u_k}^1(t_k)$ will be denoted by P_{u_k}. Recall $\vec{u}_K = (u_1, ..., u_K)$.

For K=1, the vector $\vec{u}_1$=u_1, and the following is true:

$$\max_{\vec{u}_1} \min_{\vec{\alpha}} \{P_{u_1}\} = \max_{u_1} \min_{\vec{\alpha}} \{P_{u_1}\} = P_{u_1^*},$$

so the equality in (2.10) holds.

Now assume (2.10) is true for $K = m$, so that

$$\begin{aligned}\max_{\vec{u}_m} \min_{\vec{\alpha}} \{P_{u_1} + P_{u_2}(1 - P_{u_1}) + ... + P_{u_m}(1 - P_{u_{m-1}})...(1 - P_{u_1})\} \\ = P_{u_1^*} + ... + P_{u_m^*}(1 - P_{u_{m-1}^*})...(1 - P_{u_1^*}) \\ = \max_{u_m} \min_{\vec{\alpha}} \{P_{u_1^*} + ... + P_{u_m}(1 - P_{u_{m-1}^*})...(1 - P_{u_1^*})\}. \quad (2.11)\end{aligned}$$

It must now be shown that (2.10) is true for $K = m + 1$ that is,

$$\begin{aligned}\max_{\vec{u}_{m+1}} \min_{\vec{\alpha}} \{P_{u_1} + ... + P_{u_m}(1 - P_{u_{m-1}})...(1 - P_{u_1}) \\ + P_{u_{m+1}}(1 - P_{u_m})...(1 - P_{u_1})\} \\ = P_{u_1^*} + ... + P_{u_m^*}(1 - P_{u_{m-1}^*})...(1 - P_{u_1^*}) \\ + P_{u_{m+1}^*}(1 - P_{u_m^*})...(1 - P_{u_1^*}) \\ = \max_{u_{m+1}} \min_{\vec{\alpha}} \{P_{u_1^*} + ... + P_{u_m^*}(1 - P_{u_{m-1}^*})...(1 - P_{u_1^*}) \\ + P_{u_{m+1}}(1 - P_{u_m^*})...(1 - P_{u_1^*})\}. \quad (2.12)\end{aligned}$$

Using (2.11) and (1.8), the right hand side of (2.12) becomes

$$\begin{aligned}= \max_{u_{m+1}} \min_{\vec{\alpha}} \{\max_{\vec{u}_m} \min_{\vec{\alpha}} \{P_{u_1} + ... + P_{u_m}(1 - P_{u_{m-1}})...(1 - P_{u_1})\} \\ + P_{u_{m+1}}[1 - (P_{u_1^*} + ... + P_{u_m^*}(1 - P_{u_{m-1}^*})...(1 - P_{u_1^*}))]\}.\end{aligned}$$

Using (2.11) again this becomes

$$\begin{aligned}= \max_{u_{m+1}} \min_{\vec{\alpha}} \{\max_{\vec{u}_m} \min_{\vec{\alpha}} \{P_{u_1} + ... + P_{u_m}(1 - P_{u_{m-1}})...(1 - P_{u_1})\} \\ + P_{u_{m+1}}[1 - \max_{\vec{u}_m} \min_{\vec{\alpha}} \{P_{u_1} + ... + P_{u_m}(1 - P_{u_{m-1}})...(1 - P_{u_1})\}]\}.\end{aligned}$$

Grouping terms gives

$$= \max_{u_{m+1}} \min_{\vec{\alpha}} \{ \max_{\vec{u}_m} \min_{\vec{\alpha}} \{ P_{u_1} + ... + P_{u_m}(1 - P_{u_{m-1}})...(1 - P_{u_1}) \}$$
$$\times (1 - P_{u_{m+1}}) + P_{u_{m+1}} \}.$$

Since u_{m+1} is not included in a search over $\vec{u}_m$ this becomes

$$= \max_{u_{m+1}} \min_{\vec{\alpha}} \{ \max_{\vec{u}_m} \min_{\vec{\alpha}} \{ [P_{u_1} + ... + P_{u_m}(1 - P_{u_{m-1}})...(1 - P_{u_1})]$$
$$\times (1 - P_{u_{m+1}}) + P_{u_{m+1}} \} \}.$$

Using (1.8) yields the left hand side of (2.12):

$$= \max_{\vec{u}_{m+1}} \min_{\vec{\alpha}} \{ P_{u_1} + ... + P_{u_m}(1 - P_{u_{m-1}})...(1 - P_{u_1})$$
$$+ P_{m+1}(1 - P_{u_m})...(1 - P_{u_1}) \}_{Q.E.D.}$$

Algorithm 2 is also a trimmed search. As with Algorithm 1, a total of $\sum_{k=1}^{K} \mid \mathcal{U}_k \mid$ path comparisons were made to obtain $\vec{u}_{opt}$, as opposed to a total of $\prod_{k=1}^{K} \mid \mathcal{U}_k \mid$ path comparisons required in an exhaustive search.

2.1.3 Bayesian Formulation

Suppose again that the tracking methods of the attackers are unknown to the target, but that a prior probability distribution describing the tracking method of the attackers is available. A Bayesian formulation is applied to find the best allocation scheme in this situation.

Let $\vec{\alpha}=(\alpha_1, ..., \alpha_n)$ represent the tracking method vector for n attackers. There are 2^n tracking method vectors. Let $\vec{\alpha}_j$ denote the j^{th} tracking method vector, and let π_j be the probability that $\vec{\alpha}_j$ is the true tracking method vector.

The target seeks a control vector which maximizes, over all feasible control vectors $\vec{u}_K$, the Bayes risk given by

$$R_1(\vec{u}_K)\pi_1 + R_2(\vec{u}_K)\pi_2 + ... + R_{2^n}(\vec{u}_K)\pi_{2^n}, \tag{2.13}$$

where $R_m(\vec{u}_K)$ is the conditional risk defined in Section 2.1.2. The control vector achieving this maximum is denoted by $\vec{u}_{opt}$.

The arrival state of the i^{th} attacker is given, as in Section 2.1.2, by

$$(r_i(\tau_i) = r_A, \theta_i(\tau_i), \beta_i(\tau_i) = T, \alpha_i, \tau_i),$$

where $\theta_i(\tau_i)$ and τ_i are given, and α_i is unknown.

The cost of $\vec{u}_{opt}$ is given by

$$\begin{aligned} J(\vec{u}_{opt}) &= \max_{\vec{u}_K} Pr(\text{Miss}) \\ &= \max_{\vec{u}_K}\{\sum_{j=1}^{2^n} Pr(\text{Miss} \mid \vec{\alpha}_j)\pi_j\} \\ &= \max_{\vec{u}_K}\{\sum_{j=1}^{2^n}[\prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1} P^i_{u_k}(t_k \mid \vec{\alpha}_j)(1 - P^i_{u_l}(t_l \mid \vec{\alpha}_j))]]\pi_j\} \\ &= \max_{\vec{u}_K}\{\sum_{j=1}^{2^n} R_j(\vec{u}_K)\pi_j\}. \end{aligned} \tag{2.14}$$

Determining $\vec{u}_{opt}$ by exhaustive search requires solving (2.14), which involves calculating the quantity in (2.13) for every path in the tree, and then comparing the resulting $\prod_{k=1}^{K} \mid \mathcal{U}_k \mid$ quantities to find the maximum.

Instead of exhaustively searching the set of feasible control vectors to find $\vec{u}_{opt}$, a sequential algorithm can again be applied that will result in the same $\vec{u}_{opt}$. This algorithm is similar to Algorithms 1 and 2, and differs from them only in the criterion by which it selects the survivor path,

Algorithm 3:

(1) At time t_1, apply the control that achieves

$$\max_{u_1}\{\sum_{j=1}^{2^n}[\prod_{i=1}^{n} P^i_{u_1}(t_1 \mid \vec{\alpha}_j)]\pi_j\}.$$

Denote this control by u_1^*.

(2) At time t_2, apply the control that achieves

$$\max_{u_2}\{\sum_{j=1}^{2^n}[\prod_{i=1}^{n}[P^i_{u_1^*}(t_1 \mid \vec{\alpha}_j) + P^i_{u_2}(t_2 \mid \vec{\alpha}_j)(1 - P^i_{u_1^*}(t_1 \mid \vec{\alpha}_j))]]\pi_j\}.$$

Denote this control by u_2^*.

(3) At time t_m, apply the control that achieves

$$\max_{u_m}\{\sum_{j=1}^{2^n}[\prod_{i=1}^{n}[[\sum_{k=1}^{m-1}\prod_{l=1}^{k-1} P^i_{u_k^*}(t_k \mid \vec{\alpha}_j)(1 - P^i_{u_l^*}(t_l \mid \vec{\alpha}_j))] + P^i_{u_m}(t_m \mid \vec{\alpha}_j)\prod_{l=1}^{m-1}(1 - P^i_{u_l^*}(t_l \mid \vec{\alpha}_j))]]\pi_j\}.$$

Denote this control by u_m^*.

(4) Continue until K controls have been determined. The optimal control vector is given by $\vec{u}^{A3}_{opt} = (u_1^*, ..., u_K^*)$.

The cost of $\vec{u}^{A3}_{opt}$ is given by

$$J(\vec{u}^{A3}_{opt}) = \sum_{j=1}^{2^n}[\prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1} P^i_{u_k^*}(t_k \mid \vec{\alpha}_j)(1 - P^i_{u_l^*}(t_l \mid \vec{\alpha}_j))]]\pi_j. \qquad (2.15)$$

Proof:

As in Sections 2.1.1 and 2.1.2, induction is used to show that Algorithm 3 and the exhaustive search both arrive at the same $\vec{u}_{opt}$. It is desired to show that

$J(\vec{u}_{opt})=J(\vec{u}^{A3}_{opt})$, that is to show the following:

$$\max_{\vec{u}_K}\{\sum_{j=1}^{2^n}[\prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1}P^i_{u_k}(t_k \mid \vec{\alpha}_j)(1-P^i_{u_l}(t_l \mid \vec{\alpha}_j))]]\pi_j\}$$

$$= \sum_{j=1}^{2^n}[\prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1}P^i_{u^*_k}(t_k \mid \vec{\alpha}_j)(1-P^i_{u^*_l}(t_l \mid \vec{\alpha}_j))]]\pi_j$$

$$= \max_{u_K}\{\sum_{j=1}^{2^n}[\prod_{i=1}^{n}[[\sum_{k=1}^{K-1}\prod_{l=1}^{k-1}P^i_{u^*_k}(t_k \mid \vec{\alpha}_j)(1-P^i_{u^*_l}(t_l \mid \vec{\alpha}_j))]$$

$$+P^i_{u_K}(t_K \mid \vec{\alpha}_j)\prod_{l=1}^{K-1}(1-P^i_{u^*_l}(t_l \mid \vec{\alpha}_j))]]\pi_j\}. \qquad (2.16)$$

where the $u^*_1,...,u^*_K$ are the K controls determined by Algorithm 3.

Without loss of generality, assume n=1. For notational convenience in this proof, the quantity $P^1_{u_k}(t_k \mid \vec{\alpha}_j)$ will be denoted by $P_{u_{kj}}$. Recall $\vec{u}_K=(u_1,...,u_K)$.

For K=1, the vector $\vec{u}_1=u_1$, and the following is true:

$$\max_{\vec{u}_1}\{\sum_{j=1}^{2}P_{u_{1j}}\pi_j\} = \max_{u_1}\{\sum_{j=1}^{2}P_{u_{1j}}\pi_j\} = P_{u^*_{1j}},$$

so the equality in (2.16) holds.

Now assume (2.16) is true for $K = m$, so that

$$\max_{\vec{u}_m}\{\sum_{j=1}^{2}[P_{u_{1j}}+...+P_{u_{mj}}(1-P_{u_{m-1j}})...(1-P_{u_{1j}})]\pi_j\}$$

$$= \sum_{j=1}^{2}[P_{u^*_{1j}}+...+P_{u^*_{mj}}(1-P_{u^*_{m-1j}})...(1-P_{u^*_{1j}})]\pi_j$$

$$= \max_{u_m}\{\sum_{j=1}^{2}[P_{u^*_{1j}}+...+P_{u_{mj}}(1-P_{u^*_{m-1j}})...$$

$$...(1-P_{u^*_{1j}})]\pi_j\}. \qquad (2.17)$$

It must now be shown that (2.16) is true for $K = m+1$, that is

$$\max_{\vec{u}_{m+1}}\{\sum_{j=1}^{2}[P_{u_{1j}}+...+P_{u_{mj}}(1-P_{u_{m-1j}})...(1-P_{u_{1j}})$$

$$+P_{u_{m+1\flat}}(1-P_{u_{m\flat}})...(1-P_{u_{1\flat}})]\pi_j\}$$

$$=\sum_{j=1}^{2}[P_{u^*_{1\flat}}+...+P_{u^*_{m\flat}}(1-P_{u^*_{m-1\flat}})...(1-P_{u^*_{1\flat}})$$

$$+P_{u^*_{m+1\flat}}(1-P_{u^*_{m\flat}})...(1-P_{u^*_{1\flat}})]\pi_j$$

$$=\max_{u_{m+1}}\{\sum_{j=1}^{2}[P_{u^*_{1\flat}}+...+P_{u^*_{m\flat}}(1-P_{u^*_{m-1\flat}})...(1-P_{u^*_{1\flat}})$$

$$+P_{u_{m+1\flat}}(1-P_{u^*_{m\flat}})...(1-P_{u^*_{1\flat}})]\pi_j\}. \qquad (2.18)$$

Using (2.17) and (1.8), the right hand side of (2.18) becomes

$$=\max_{u_{m+1}}\{\max_{\vec{u}_m}\{\sum_{j=1}^{2}[P_{u_{1\flat}}+...+P_{u_{m\flat}}(1-P_{u_{m-1\flat}})...(1-P_{u_{1\flat}})]\pi_j\}$$

$$+\sum_{j=1}^{2}P_{u_{m+1\flat}}[1-(P_{u^*_{1\flat}}+...+P_{u^*_{m\flat}}(1-P_{u^*_{m-1\flat}})...(1-P_{u^*_{1\flat}}))]\pi_j\}.$$

Using (2.17) again this becomes

$$=\max_{u_{m+1}}\{\max_{\vec{u}_m}\{\sum_{j=1}^{2}[P_{u_{1\flat}}+...+P_{u_{m\flat}}(1-P_{u_{m-1\flat}})...(1-P_{u_{1\flat}})]\pi_j\}$$

$$+\sum_{j=1}^{2}P_{u_{m+1\flat}}[\pi_j-\max_{\vec{u}_m}\{\sum_{j=1}^{2}[P_{u_{1\flat}}+...$$

$$...+P_{u_{m\flat}}(1-P_{u_{m-1\flat}})...(1-P_{u_{1\flat}})]\pi_j\}]\}.$$

Grouping terms gives

$$=\max_{u_{m+1}}\{\max_{\vec{u}_m}\{\sum_{j=1}^{2}[P_{u_{1\flat}}+...+P_{u_{m\flat}}(1-P_{u_{m-1\flat}})...(1-P_{u_{1\flat}})]\pi_j\}$$

$$\times(1-\sum_{j=1}^{2}P_{u_{m+1\flat}})+\sum_{j=1}^{2}P_{u_{m+1\flat}}\pi_j\}.$$

Since u_{m+1} is not included in a search over $\vec{u}_m$ this becomes

$$=\max_{u_{m+1}}\{\max_{\vec{u}_m}\{[\sum_{j=1}^{2}[P_{u_{1\flat}}+...+P_{u_{m\flat}}(1-P_{u_{m-1\flat}})...(1-P_{u_{1\flat}})]\pi_j]$$

$$\times(1-\sum_{j=1}^{2}P_{u_{m+1\flat}})+\sum_{j=1}^{2}P_{u_{m+1\flat}}\pi_j]\}\}.$$

Using (1.8) yields the left hand side of (2.18):

$$= \max_{\vec{u}_{m+1}} \{ \sum_{j=1}^{2} [P_{u_{1\mathfrak{b}}} + ... + P_{u_{m\mathfrak{b}}}(1 - P_{u_{m-1\mathfrak{b}}})...(1 - P_{u_{1\mathfrak{b}}}) + P_{u_{m+1\mathfrak{b}}}(1 - P_{u_{m\mathfrak{b}}})...(1 - P_{u_{1\mathfrak{b}}})]\pi_j \}_{Q.E.D.}$$

Algorithm 3 is again a trimmed search. As with Algorithms 1 and 2, a total of $\sum_{k=1}^{K} \mid \mathcal{U}_k \mid$ path comparisons are made to obtain $\vec{u}_{opt}$, as opposed to $\prod_{k=1}^{K} \mid \mathcal{U}_k \mid$ path comparisons required in an exhaustive search.

Note that since the set of predicted feasible control vectors is defined a priori, the optimal allocation scheme in each formulation can be determined a priori.

Algorithms 1, 2, and 3, were applied to several attack scenarios, and the resulting optimal control vectors and corresponding probabilities of target survival are given in Chapter 5.

2.2 States Updates Available

Now suppose updates of the attackers states will be provided to the target every Δ time units. When a decision instant occurs, the target uses the update, and the knowledge of the previously applied controls, to determine the set of controls currently feasible.

Obtaining the control vector which maximizes the quantities in (2.2), (2.8), and (2.14) involves exhaustively searching over a set of feasible control vectors of dimension K. However, because the controls feasible at a given decision instant will not be known until that decision instant actually occurs, this set cannot be completely determined until time t_K. Therefore $\vec{u}_{opt}$ cannot be determined prior to the first arrival, and the optimal control to apply at each decision instant

must be determined at the time the decision instant occurs.

However, if a set of feasible control vectors is defined at every decision instant, it is possible to use exhaustive search to determine the optimal allocation scheme in each formulation element by element. This requires defining a total of K sets of feasible control vectors. The members of each set depend on the previously applied controls, the currently feasible controls, and the controls predicted to be feasible at the remaining future decision instants. The predicted controls are determined as in Section 2.1, by predicting the future states of the attackers. The set of predicting equations for this case takes advantage of the updates, and is slightly different from the set of predicting equations given in (2.1). At the k^{th} decision instant, if the i^{th} attacker is in the tracking state, its future states are predicted according to the following equations:

$$\begin{aligned} r_i(t) &= r_A - v_A(t-\tau_i) \; ; \; t_k < t < t_K \\ \theta_i(t) &= \theta_i(\tau_i) \; ; \; t_k < t < t_K \\ \beta_i(t) &= T \; ; \; t_k < t < t_K. \end{aligned} \tag{2.19}$$

If the i^{th} attacker is in the not-tracking state at time t_k, it is no longer a threat to the target as it cannot transition back into the T state, and it is not necessary to predict its future states.

The set of feasible control vectors defined for the k^{th} decision instant contains a total of $\prod_{i=k}^{K} \mid \mathcal{U}_i \mid$ members. The first $k-1$ controls are fixed. Denote these controls by $u_1^*, ..., u_{k-1}^*$. The set $\mathcal{U}_k$ contains the controls currently feasible. The sets $\mathcal{U}_i$ for $i = k+1, ..., K$ contain the controls predicted to be feasible at times $t_{k+1}, ..., t_K$.

In the deterministic formulation, where the tracking methods of the attackers

are known a priori, at time t_k this set of vectors is searched to find the vector which achieves

$$\max_{u_k,\ldots,u_K}\{\prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1}P^i_{u_k}(t_k)(1-P^i_{u_l}(t_l))]\}, \tag{2.20}$$

where $u_i = u_i^*$ for $i = 1,\ldots,k-1$. The element in the k^{th} position of the resulting vector is the control applied at time t_k. This control is denoted by u_k^* and is the element in the k^{th} position of $\vec{u}_{opt}$. The $\vec{u}_{opt}$ determined by this method is given by $\vec{u}_{opt} = (u_1^*,\ldots,u_K^*)$.

In the minimax formulation, where the tracking methods of the attackers are totally unknown, and there is no prior probability distribution available describing this parameter, at time t_k the set of vectors is searched to find the vector which achieves

$$\max_{u_k,\ldots,u_K}\min_{\vec{\alpha}}\{[\prod_{i=1}^{n}[\sum_{k=1}^{K}\prod_{l=1}^{k-1}P^i_{u_k}(t_k)(1-P^i_{u_l}(t_l))]]\}, \tag{2.21}$$

where $u_i = u_i^*$ for $i = 1,\ldots,k-1$. The element in the k^{th} position of the resulting vector is the control applied at time t_k. This control is denoted by u_k^* and is the element in the k^{th} position of $\vec{u}_{opt}$. The $\vec{u}_{opt}$ determined by this method is given by $\vec{u}_{opt} = (u_1^*,\ldots,u_K^*)$.

In the Bayesian formulation, where the tracking methods of the attackers are again totally unknown, but there is a prior probability distribution available describing this parameter, at time t_k the set of vectors is searched to find the vector which achieves

$$2^n \quad n \quad K \quad k-1$$

element in the k^{th} position of $\vec{u}_{opt}$. The $\vec{u}_{opt}$ determined by this method is given by $\vec{u}_{opt} = (u_1^*, ..., u_K^*)$.

In all three formulations, after the last control u_K^* is determined, each exhaustive search approach will have made a total of $\sum_{m=1}^{K} \prod_{k=m}^{K} \mid \mathcal{U}_k \mid$ path comparisons to obtain $\vec{u}_{opt}$.

Applying Algorithms 1, 2, and 3 will yield the same $\vec{u}_{opt}$ as their respective exhaustive search approaches. It was shown in Section 2.1.1., that for the same set of feasible control vectors, the optimal control vector determined by Algorithm 1 was equal to the optimal control vector determined by solving (2.2). Therefore the K control vectors that would result from solving (2.20) at each decision instant, will equal the K control vectors that would be determined by applying Algorithm 1 at each decision instant. Similarly, it was shown in Sections 2.1.2 and 2.1.3, that Algorithm 2 and Algorithm 3 yield the same optimal allocation scheme as their respective exhaustive searches. Therefore the K control vectors that would result from solving (2.21) and (2.22) at each decision instant, will equal the K control vectors that would be determined by applying Algorithm 2 and Algorithm 3 at each decision instant.

However, it is not necessary to define a set of feasible control vectors at each decision instant for Algorithms 1, 2, and 3 to find the optimal controls. At time t_k, each algorithm selects u_k^* from the set of controls currently feasible, using only the knowledge of the states of the attackers at times $t_1, ..., t_k$, and knowledge of the controls applied at times $t_1, ..., t_{k-1}$.

In the deterministic formulation, at time t_k Algorithm 1 applies the control

that achieves the following

$$\max_{u_k \in \mathcal{U}_k} \{\prod_{i=1}^{n} [\sum_{m=1}^{k-1} \prod_{l=1}^{m-1} P^i_{u^*_m}(t_m)(1 - P^i_{u^*_l}(t_l))] + P^i_{u_k}(t_k) \prod_{l=1}^{k-1} (1 - P^i_{u^*_l}(t_l))\}, \quad (2.23)$$

where the u^*_i, i=1,...,k-1 are the controls that were applied at the first $k-1$ decision instants.

In the minimax formulation, at time t_k Algorithm 2 applies the control that achieves the following

$$\max_{u_k \in \mathcal{U}_k} \min_{\vec{\alpha}} \{\prod_{i=1}^{n} [\sum_{m=1}^{k-1} \prod_{l=1}^{m-1} P^i_{u^*_m}(t_m)(1 - P^i_{u^*_l}(t_l))] + P^i_{u_k}(t_k) \prod_{l=1}^{k-1} (1 - P^i_{u^*_l}(t_l))\}, \quad (2.24)$$

where the u^*_i, i=1,...,k-1 are the controls that were applied at the first $k-1$ decision instants.

In the Bayesian formulation, at time t_k Algorithm 3 applies the control that achieves the following

$$\max_{u_k \in \mathcal{U}_k} \{\sum_{j=1}^{2^n} [\prod_{i=1}^{n} [[\sum_{m=1}^{k-1} \prod_{l=1}^{m-1} P^i_{u^*_m}(t_m \mid \vec{\alpha}_j)(1 - P^i_{u^*_l}(t_l \mid \vec{\alpha}_j))] + P^i_{u_k}(t_k \mid \vec{\alpha}_j) \prod_{l=1}^{k-1} (1 - P^i_{u^*_l}(t_l \mid \vec{\alpha}_j))]] \pi_j\}, \quad (2.25)$$

where the u^*_i, $i = 1, ..., k-1$ are the controls that were applied at the first $k-1$ decision instants.

Algorithms 1, 2, and 3, have two advantages over their respective exhaustive search approaches. The first is that they do not require determining K sets of feasible control vectors at each decision instant, which involves predicting future attacker states, in order to determine the $\vec{u}_{opt}$. The second advantage is a consequence of the first. After the last control u^*_K is determined, the exhaustive search approaches will have made a total of $\sum_{m=1}^{K} \prod_{k=m}^{K} \mid \mathcal{U}_k \mid$ path comparisons.

In contrast, Algorithms 1, 2, and 3 will have made only $\sum_{k=1}^{K} \mid \mathcal{U}_k \mid$ path comparisons.

Chapter 3

Unknown Arrival Times

Now suppose neither the arrival times nor the arrival angles of the attackers are known to the target a priori. As a result, the number of attackers arriving in the interval $[0,\Omega]$ is not known.

In the case of unavailable updates, finding the $\vec{u}_{opt}$ in each of the three formulations by exhaustive search requires knowledge of some set of feasible control vectors. If the arrival schedule is not known a priori, the target will never know if any attackers have arrived. This results in an infinite number of feasible control vectors, and it makes finding the optimal control vectors by exhaustive search impossible.

If at each decision instant, the target is provided with an update of the states of all the attackers present, it can use this information and knowledge of the previously applied controls to determine the set of controls currently feasible. However, if the arrival times are not known a priori, it is not possible to predict the controls that will be feasible at future decision instants by the method in Section 2.2, and using exhaustive search to determine $\vec{u}_{opt}$ in each of the three formulations element by element, as in Section 2.2, is not possible.

Hence the exhaustive search approach cannot be used to determine the optimal control vector in any of the formulations when the arrival schedule is not known prior to the first attacker's arrival. However, using the probability of target survival in (1.11), Algorithm 2 can be applied to determine $\vec{u}_{opt}$ in the minimax formulation, as long as updates of the attackers states are provided at each decision instant.

Recall a decoy deployed at the r^{th} decision instant can affect the tracking state of any attacker that arrives in the time interval $[t_r, t_r + \mu_D]$. If the target is considering deploying a decoy at time t_r, it is not necessary to know the exact arrival times of each attacker that will arrive in $[t_r, t_r + \mu_D]$, but only the number of attackers that will arrive in $[t_r, t_r + \mu_D]$, and their arrival angles, in order to determine the best control to apply at t_r. Since the arrival times are unknown, the number of attackers that will arrive in any time interval is unknown. Redefine n to be a Poisson random variable describing the number of attackers arriving in an interval of length μ_D. Let n_r denote the number of attackers present at time t_r.

As described in the introduction, the arrival angle of an attacker can be in

any one of M sectors with equal probability. Let $\vec{\theta}_m = (\theta_{n_r+1}, ..., \theta_{n_r+m})$ represent the arrival angle vector for m attackers arriving in $[t_r, t_r + \mu_D]$, where $\theta_i \in \{S_1, ..., S_M\}$. There are M^m possible arrival angle vectors. Let $\vec{\theta}_{m,y}$ denote the y^{th} arrival angle vector of dimension m. Let $\vec{\alpha}_{n_r+m}=(\alpha_1, ..., \alpha_{n_r}, \alpha_{n_r+1}, ..., \alpha_{n_r+m})$, where $\alpha_i \in \{LE, CEN\}$. There are 2^{n_r+m} possible tracking method vectors.

The probability of target survival is written as

$$
\begin{aligned}
&Pr(\text{Miss}) \\
&\quad = \sum_{m=1}^{\infty} Pr(\text{Miss} \mid m \text{ attackers arrive in an interval of length } \mu_D) \\
&\qquad\qquad \times Pr(m \text{ attackers arrive in an interval of length } \mu_D) \\
&\quad = \sum_{m=1}^{\infty} Pr(\text{Miss} \mid \text{n} = m) Pr(\text{n} = m) \\
&\quad = \sum_{m=1}^{\infty} \sum_{y=1}^{M^m} Pr(\text{Miss} \mid \text{n} = m, \vec{\theta}_{m,y}) Pr(\vec{\theta}_{m,y}) Pr(\text{n} = m). \qquad (3.1)
\end{aligned}
$$

Since the attacker arrivals are independent, and the arrival rate is stationary, (3.1) becomes

$$
\sum_{m=1}^{\infty} \sum_{y=1}^{M^m} Pr(\text{Miss} \mid \text{n} = m, \vec{\theta}_{m,y}) \frac{1}{M^m} \frac{(\lambda \mu_D)^m e^{-\lambda \mu_D}}{m!}. \qquad (3.2)
$$

At time t_r, the target first determines the set of controls currently feasible $\mathcal{U}_r$, and then must evaluate the following expression in order to determine u_r^*:

$$
\begin{aligned}
\max_{u_r} \min_{\vec{\alpha}} \{ &\sum_{m=1}^{\infty} \sum_{y=1}^{M^m} \frac{1}{M^m} \frac{(\lambda \mu_D)^m e^{-\lambda \mu_D}}{m!} \\
&\times [\prod_{i=1}^{n_r+m} [\sum_{k=1}^{r} \prod_{l=1}^{k-1} P^i_{u_k}(t_k \mid \vec{\theta}_{m,y})(1 - P^i_{u_l}(t_l \mid \vec{\theta}_{m,y}))]]\}, \qquad (3.3)
\end{aligned}
$$

where $u_i = u_i^*$ for $i = 1, ..., r-1$. There is not a closed form expression for (3.3), and there are an infinite number of terms. This expression will be approximated by truncating the sum at the term for which the value of m is such that $Pr(\text{n} =$

$m) \leq \epsilon$. Denote this value of m by $\hat{m}$. The quantity evaluated by Algorithm 2 at t_r is then given by

$$\max_{u_r} \min_{\vec{\alpha}_m} \{ \sum_{m=1}^{\hat{m}} \sum_{y=1}^{M^m} \frac{1}{M^m} \frac{(\lambda \mu_D)^m e^{-\lambda \mu_D}}{m!} \times [\prod_{i=1}^{n_r+m} [\sum_{k=1}^{r} \prod_{l=1}^{k-1} P^i_{u_k}(t_k \mid \vec{\theta}_{m,y})(1 - P^i_{u_l}(t_l \mid \vec{\theta}_{m,y}))]]\}. \quad (3.4)$$

where $u_i = u_i^*$ for $i = 1, \ldots, r-1$. The control that achieves the maximum is denoted by u_r^* and is applied at time t_r.

It must be shown that the solution u_r^* to (3.4) is the same control as would be determined if the target knew that exactly $\hat{n}$ attackers would arrive in $[t_r, t_r + \mu_D]$, for $1 \leq \hat{n} \leq \hat{m}$. This can be shown by induction.

The idea behind the proof is to show that if the maximum of a sum of functions is equal to the sum of the maxima, then each function is maximized at the same point.

Proof:

For notational convenience in this proof, the following quantities are defined:

$$\sum_{k=1}^{r} \prod_{l=1}^{k-1} P^i_{u_k}(t_k \mid \vec{\theta}_{m,y})(1 - P^i_{u_l}(t_l \mid \vec{\theta}_{m,y})) = Q^i_y$$

$$\frac{1}{M^m} \frac{(\lambda \mu_D)^m e^{-\lambda \mu_D}}{m!} = \gamma_m.$$

The expression in (3.4) then becomes:

$$\max_{u_r} \min_{\vec{\alpha}_{m+n_r}} \{ \sum_{m=1}^{\hat{m}} \sum_{y=1}^{M^m} [\prod_{i=1}^{n_r+m} Q^i_y \gamma_m]\}.$$

Without loss of generality, assume $M=1$ and $n_r=1$. It is desired to show the following:

$$\max_{u_r} \min_{\vec{\alpha}_{m+1}} \{ \sum_{m=1}^{\hat{m}} \prod_{i=1}^{1+m} Q^i \gamma_m \} = \sum_{m=1}^{\hat{m}} \gamma_m \max_{u_r} \min_{\vec{\alpha}_{m+1}} \{ \prod_{i=1}^{1+m} Q^i \}. \quad (3.5)$$

For $\hat{m}$=1, there is only one term in the sum,

$$\max_{u_r} \min_{\vec{\alpha}_2} \{Q^1 Q^2 \gamma_1\} \quad = \quad \gamma_1 \max_{u_r} \min_{\vec{\alpha}_2} \{Q^1 Q^2\},$$

so the equality in (3.5) holds.

Now assume (3.5) holds for $\hat{m}$=p, so that

$$\begin{aligned} &\max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 Q^2 \gamma_1 + ... + Q^1 ... Q^{p+1} \gamma_p\} \\ &\quad = \gamma_1 \max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 Q^2\} + ... + \gamma_p \max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 ... Q^{p+1}\}. \end{aligned} \tag{3.6}$$

It must now be shown that (3.5) is true for $\hat{m}$=$p+1$, that is

$$\begin{aligned} &\max_{u_r} \min_{\vec{\alpha}_{p+2}} \{Q^1 Q^2 \gamma_1 + ... + Q^1 ... Q^{p+1} \gamma_p + Q^1 ... Q^{p+2} \gamma_{p+1}\} \\ &\quad = \gamma_1 \max_{u_r} \min_{\vec{\alpha}_{p+2}} \{Q^1 Q^2\} + ... + \gamma_p \max_{u_r} \min_{\vec{\alpha}_{p+2}} \{Q^1 ... Q^p\} \\ &\qquad + \gamma_{p+1} \max_{u_r} \min_{\vec{\alpha}_{p+2}} \{Q^1 ... Q^{p+2}\}. \end{aligned} \tag{3.7}$$

The left hand side of (3.7) can be written as

$$\begin{aligned} &= \max_{u_r} \min_{\alpha_{p+2}} \{\max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 Q^2 \gamma_1 + ... + Q^1 ... Q^{p+1} \gamma_p + Q^1 ... Q^{p+2} \gamma_{p+1}\}\} \\ &= \max_{u_r} \min_{\alpha_{p+2}} \{\max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 Q^2 \gamma_1 + ... + Q^1 ... Q^{p+1} (\gamma_p + Q^{p+2} \gamma_{p+1})\}\}. \end{aligned}$$

Since α_{p+2}, and thus Q^{p+2}, is not included in a search over $\vec{\alpha}_{p+1}$, using (3.6) yields the right hand side of (3.7):

$$\begin{aligned} &= \max_{u_r} \min_{\alpha_{p+2}} \{\gamma_1 \max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 Q^2\} + ... \\ &\qquad ... + (\gamma_p + Q^{p+2} \gamma_{p+1}) \max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 ... Q^{p+1}\}\} \\ &= \max_{u_r} \min_{\alpha_{p+2}} \{\gamma_1 \max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 Q^2\} + ... \\ &\qquad ... + \gamma_p \max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 ... Q^{p+1}\} + \gamma_{p+1} \max_{u_r} \min_{\vec{\alpha}_{p+1}} \{Q^1 ... Q^{p+2}\}\} \\ &= \gamma_1 \max_{u_r} \min_{\vec{\alpha}_{p+2}} \{Q^1 Q^2\} + ... + \gamma_p \max_{u_r} \min_{\vec{\alpha}_{p+2}} \{Q^1 ... Q^p\} \\ &\qquad + \gamma_{p+1} \max_{u_r} \min_{\vec{\alpha}_{p+2}} \{Q^1 ... Q^{p+2}\}_{Q.E.D.} \end{aligned}$$

The length of the interval $[0,\Omega]$ must be defined. Time zero is considered to be the time at which the target is first alerted to the presence of an attacker. The time Ω can be defined in two ways. The first way is to set Ω to some predefined value. The second way is to consider Ω the moment at which some predefined amount of time has elapsed since the last arrival.

Chapter 4

Neural Network

Hopfield and Tank [1] have developed a neural network that has been shown to be effective in solving optimization problems by mapping them to analog computation.

A biological nervous system makes decisions rapidly by operating collectively in a parallel analog mode. Hopfield and Tank designed a simple analog electronic nervous system, comprised of a matrix of nonlinear analog elements, which exhibits the essential features of a biological nervous system: parallel inputs, parallel outputs, and extensive interconnectivity between the electronic neurons.

A circuit schematic of the network is given in Figure 4.1. Each neuron is represented by a pair of amplifiers with a parallel RC circuit at a common input. The output of a neuron is fed back to its input and can also be connected to the input of any other neuron. There is an externally supplied current into each neuron.

The RC circuit helps define the time constant of a neuron, and allows for integration of all the input currents entering the neuron from other neurons in the network. The synapse between two neurons is represented by the connection of the output of one of the two amplifiers of a neuron to the input of another amplifier pair through a resistor. The resistor connecting the i^{th} and j^{th} neuron is denoted by T_{ij}. If the synapse is inhibitory, the output of the inverting amplifier is connected to the resistor. If the synapse is excitatory, the output of the non-inverting amplifier is connected to the resistor. The magnitude of the synapse is determined by the value of the resistor. The external input current into each neuron adjusts the general excitability of the neuron.

The voltage into the j^{th} amplifier pair is denoted by x_j, and the output of the j^{th} non-inverting amplifier is denoted by V_j. The input/output relationship is given by $V_j = \frac{1}{2}(1 + \tanh(\frac{x_j}{u_o})) = g(x_j)$, and is plotted in Figure 4.2.

Suppose there is a total of N neurons in the network. The equations describing the dynamics of the i^{th} neuron are given by

$$\begin{aligned} \frac{dx_i}{dt} &= -\frac{x_i}{RC} + \sum_{j=1}^{N} T_{ij} V_j + I_i \\ V_j &= g(x_j), \end{aligned} \tag{4.1}$$

where RC is the time constant of the RC circuit at the input of each neuron.

In an earlier paper by Hopfield [2], it was shown that under a symmetric

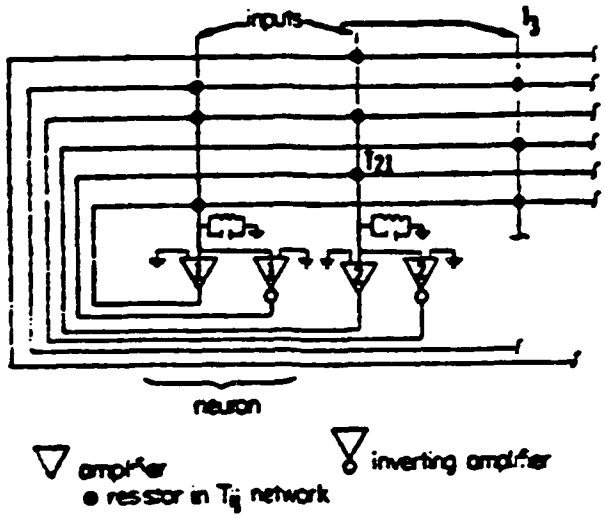

Figure 4.1: Layout of neural circuit.

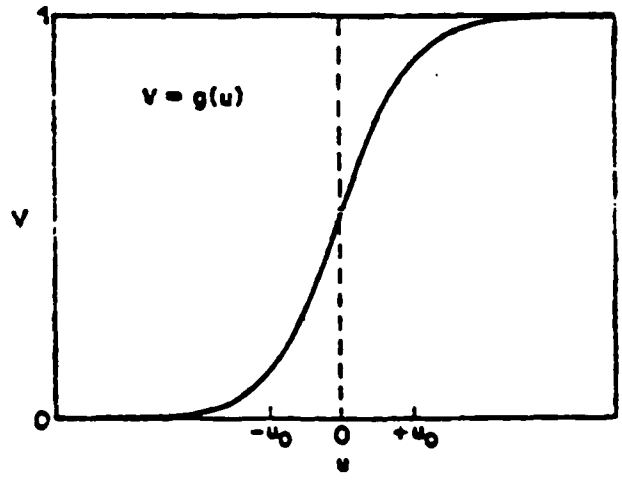

Figure 4.2: Gain curve.

connection matrix ($T_{ij} = T_{ji}$), the outputs of the neurons converge to a constant and stable state. He also showed that if the rise times of the amplifiers are very short, so that the width of the gain curve in Figure 4.2 is narrow, the stable states of the network locally minimize the following quantity:

$$E_1 = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} T_{ij}V_iV_j - \sum_{i=1}^{N} V_iI_i. \tag{4.2}$$

The state space is the interior and border of an N dimensional hypercube. Since the width of the gain curve is narrow, V_j is ultimately either 0 or 1, and the minimum of (4.2) occurs at one of the 2^N corners of the hypercube.

The allocation problem must be mapped onto this network. The first step is to choose a representation for the output. Let $N = K\sum_{k=1}^{K} \mid \mathcal{U}_k \mid$, where K is the number of decision instants. Arrange the neurons in a $\sum_{k=1}^{K} \mid \mathcal{U}_k \mid \times K$ matrix. Each row represents a control and each column represents a decision instant. The output of the neuron in the X^{th} row and j^{th} column of the matrix is given by V_{Xj}. The dynamics of the $(X,j)^{th}$ neuron are then written as:

$$\frac{dx_{Xi}}{dt} = -\frac{x_{Xi}}{RC} + \sum_{Y}\sum_{j} T_{Xi,Yj}V_{Yj} + I_{Xi}, \tag{4.3}$$

and (4.2) is written as:

$$E_1 = -\frac{1}{2}\sum_{X}\sum_{Y}\sum_{i}\sum_{j} T_{Xi,Yj}V_{Xi}V_{Yj} - \sum_{X}\sum_{i} V_{Xi}I_{Xi}. \tag{4.4}$$

When the network stabilizes, the optimal control to apply at time t_k will be determined by the values of the neurons in the k^{th} column of the matrix. For example, suppose K=3 and $\mathcal{U}_1$={L,R,Z}, $\mathcal{U}_2 = \{L, R, Z\}$, and $\mathcal{U}_3 = \{G_1, Z\}$. The output matrix shown below gives $\vec{u}_{opt}$=(L, L, G_1).

	1	2	3
L	1	0	0
R	0	0	0
Z	0	0	0
L	0	1	0
R	0	0	0
Z	0	0	0
G_1	0	0	1
Z	0	0	0

Note there is only one 1 in each column and no more than one 1 in each row for a total of K 1's in the matrix, so that at every decision instant one and only one of the controls feasible at that decision instant is selected.

The next step is to determine a function describing the network that when minimized results in exactly one control being selected for every decision instant, and also yields the optimal allocation scheme. A function meeting the first requirement is given by Hopfield and Tank [1]:

$$E_2 = \frac{A}{2}\sum_X\sum_i\sum_{j\neq i} V_{Xi}V_{Xj} + \frac{B}{2}\sum_i\sum_X\sum_{X\neq Y} V_{Xi}V_{Yi} + \frac{C}{2}(\sum_X\sum_i V_{Xi} - K)^2, \tag{4.5}$$

where A, B, and C are large positive constants. The first term is minimized when there is no more than one 1 in each row. The second term is minimized when there is only one 1 in each column. The third term is minimized when a total of K 1's appear in the matrix.

The second requirement is met by appending a term to (4.5) that represents the probability that the target does not survive the attacks. From (1.2), (1.3), and (1.5), the probability the target does not survive is given by:

$$\begin{aligned} & Pr(\text{Target does not survive}) \\ & \quad = 1 - Pr(\text{Target survives}) \\ & \quad = 1 - \prod_{l=1}^{n}[1 - Pr(\mathrm{H_l} \mid \mathrm{M_{l-1}}...\mathrm{M_l})] \\ & \quad = 1 - \prod_{l=1}^{n}(1 - \prod_{k=1}^{K}[1 - P^{l}_{u_k}(t_k)]). \end{aligned} \tag{4.6}$$

In order to discourage the network from the selection of some controls, and encourage the selection of others, the notion of a "distance" between the controls must be defined, and these distances reflected in the connections (T_{X_i,Y_j}'s) between the neurons. This can be accomplished by isolating the probability of failure of each control against all the attackers, so that the "distance" from one control to another is the probability of failure against all the attackers of applying that control. This is done by upper bounding the the expression for the probability that the target does not survive in (4.6).

Define a set $S=\{b_1, b_2, b_3, b_4 : 0 \leq b_i \leq 1,\ i = 1,2,3,4\}$. The following relations hold:

$$(i) \qquad b_1 b_2 + b_3 b_4 \leq b_1 + b_2 + b_3 + b_4$$

$$(ii) \qquad b_1 b_2 \leq 1.$$

Using these relations, (4.6) can be upperbounded as

$$Pr(\text{Target does not survive})$$

$$
\begin{aligned}
&\leq 1 - \sum_{l=1}^{n}(1 - \sum_{k=1}^{K}[1 - P^l_{u_k}(t_k)]) \\
&= 1 - n + nK - \sum_{l=1}^{n}\sum_{k=1}^{K} P^l_{u_k}(t_k) \\
&= 1 - n + \sum_{l=1}^{n}\sum_{k=1}^{K}(1 - P^l_{u_k}(t_k)) \\
&\leq \sum_{l=1}^{n}\sum_{k=1}^{K}(1 - P^l_{u_k}(t_k)) \\
&= \sum_{k=1}^{K}\sum_{l=1}^{n}(1 - P^l_{u_k}(t_k)). \qquad (4.7)
\end{aligned}
$$

Define the quantity d_{XY} as the sum of the probabilities of failure against each attacker of the control in the Y^{th} row applied at time t_Y, given the control in the X^{th} row was the last control applied. That is,

$$d_{XY} = \sum_{l=1}^{n}(1 - P^l_{Y|X}(t_Y)), \qquad (4.8)$$

where $t_Y = t_k$ if $Y \in \mathcal{U}_k$. This quantity represents the "distance" between the controls in rows X and Y. A control must be applied at each decision instant, even if it is the "do nothing" control. In other words, no decision instants can be skipped. Therefore it is required that $P^l_{Y|X}(t_Y)$=0 for l=1,...,n if $X \in \mathcal{U}_k$, but $Y \notin \mathcal{U}_{k+1}$. Also since no more than one control can be applied at each decision instant, it is required that $P^l_{Y|X}(t_Y)$=0 for $l = 1,...,n$ if $X, Y \in \mathcal{U}_k$. This is summarized below:

$$d_{XY} = \begin{cases} \sum_{l=1}^{n}(1 - P^l_{Y|X}(t_Y)) & \text{if } X \in \mathcal{U}_k \text{ and } Y \in \mathcal{U}_{k+1} \\ n & \text{otherwise.} \end{cases} \qquad (4.9)$$

From these quantities, a symmetric distance matrix ($d_{XY} = d_{YX}$) is constructed. This matrix is denoted by $\mathbf{d}$.

The following term, given by Hopfield and Tank [1], appended to (4.5) assures network converges to the scheme with the minimum cost, (or minimum "length"):

$$\frac{D}{2}\sum_X\sum_{Y\neq X}\sum_i d_{XY}V_{Xi}(V_{Y,i+1}+V_{Y,i-1}). \tag{4.10}$$

It yields the numeric value of (4.7) for the allocation scheme represented by the states of the neurons in the stabilized network.

The function describing the allocation network is the sum of (4.5) and (4.10):

$$\begin{aligned} E_2 &= \frac{A}{2}\sum_X\sum_i\sum_{j\neq i}V_{Xi}V_{Xj}+\frac{B}{2}\sum_i\sum_X\sum_{X\neq Y}V_{Xi}V_{Yi} \\ &\quad +\frac{C}{2}(\sum_X\sum_i V_{Xi}-K)^2 \\ &\quad +\frac{D}{2}\sum_X\sum_{Y\neq X}\sum_i d_{XY}V_{Xi}(V_{Y,i+1}+V_{Y,i-1}). \end{aligned} \tag{4.11}$$

When this function is minimized, the states of the neurons in the network will represent the optimal allocation scheme.

It is known that the network minimizes

$$E_1 = -\frac{1}{2}\sum_X\sum_Y\sum_i\sum_j T_{Xi,Yj}V_{Xi}V_{Yj}-\sum_X\sum_i V_{Xi}I_{Xi}. \tag{4.12}$$

The function E_1, and the function that is desired to be minimized E_2, can be made equivalent by choosing the appropriate set of $T_{Xi,Yj}$'s and I_{Xi}'s. Hopfield and Tank [1] have determined that the following connection matrix and external input currents make (4.11) equivalent to (4.12):

$$I_{Xi} = CK \tag{4.13}$$

$$\begin{aligned} T_{Xi,Yj} &= -A\delta_{XY}(1-\delta_{ij})-B\delta_{ij}(1-\delta_{XY})-C \\ &\quad -Dd_{XY}(\delta_{j,i+1}+\delta_{j,i-1}) \end{aligned} \tag{4.14}$$

$$\delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

Substituting these into (4.3), the equations describing the dynamics of the neuron in the X^{th} row and j^{th} column are given by

$$\frac{dx_{Xi}}{dt} = -\frac{x_{Xi}}{RC} - A\sum_{j\neq i} V_{Xj} - B\sum_{Y\neq X} V_{Yi} - C(\sum_{X}\sum_{j} V_{Xj} - K) - D\sum_{Y} d_{XY}(V_{Y,i+1} + V_{Y,i-1}) \quad (4.15)$$

$$V_{Xi} = g(x_{Xi}). \quad (4.16)$$

The derivations of (4.11) from (4.12) and (4.15) from (4.3), using the quantities in (4.13) and (4.14), are given in Appendix A.

4.1 Deterministic Formulation

This network can be used to obtain the optimal allocation scheme when the arrival times, arrival angles, and tracking methods of the attackers are known to the target a priori.

4.1.1 State Updates Unavailable

Suppose no updates of the attackers' states will be provided. The only attacker knowledge provided to the target is the a priori knowledge of the arrival states. The arrival states are given as in Section 2.1.1, by

$$(r_i(\tau_i) = r_A, \theta_i(\tau_i), \beta_i(\tau_i) = T, \alpha_i, \tau_i),$$

where $\theta_i(\tau_i)$, τ_i, and α_i are given.

Using the expression for the cost function given in (1.5), it is desired to determine the control vector which achieves

$$\max_{\bar{u}_K}\{\prod_{i=1}^{n}(1-\prod_{k=1}^{K}[1-P^i_{u_k}(t_k)])\}. \tag{4.17}$$

The set of feasible control vectors necessary to solve (4.17) is determined as in Section 2.1.

Since the control applied at time t_k must be a member of the set of controls feasible at t_k, the network is not allowed to consider states in which this requirement is not met. In other words, the network is only allowed to consider the feasible control vectors. To guarantee this, any neuron whose position in the matrix is such that an output of 1 would violate the above requirement, is fixed at 0. Therefore $V_{Xi}(t)$=0 for all t, if $X \in \mathcal{U}_k$ and $i \neq k$.

When the network stabilizes, the control to apply at the k^{th} decision instant is determined by the value of the neurons in the the k^{th} column. This network was used to determine the optimal allocation schemes for the same attack scenarios as considered in Section 2.1. The results are presented in Chapter 5.

4.2 Choosing Values for A, B, C, and D

Hopfield and Tank do not provide any guidelines for choosing the values of the scalars A, B, C, and D in (4.11), other than the fact that they must be positive and much greater than one. It is desired to determine relationships between these scalars which result in choosing values that help the network converge to a stable state. This is done by analyzing the network at steady state.

Suppose there is a square $\eta \times \eta$ matrix of neurons. The dynamics of a single

neuron in the matrix are given by (4.15) and (4.16):

$$\frac{dx_{Xi}}{dt} = -\frac{x_{Xi}}{RC} - A\sum_{j\neq i} V_{Xj} - B\sum_{Y\neq X} V_{Yi} - C(\sum_X \sum_j V_{Xj} - \eta) - D\sum_Y d_{XY}(V_{Y,i+1} + V_{Y,i-1}) \quad (4.18)$$

$$V_{Xi} = g(x_{Xi}). \quad (4.19)$$

Now approximate the input-output relationship of the neurons by:

$$V_{Xi} = \begin{cases} 0 & \text{if } x_{Xi} < -u_o \\ \frac{1}{2} + \frac{x_{Xi}}{2u_o} & \text{if } -u_o \leq x_{Xi} \leq u_o \\ 1 & \text{if } x_{Xi} > u_o. \end{cases}$$

Since the width of the gain curve in Figure 4.2 is assumed narrow, this approximation is not unreasonable.

Replacing $-x_{Xi}$ in (4.18) with the approximation $2u_o(\frac{1}{2} - V_{Xi})$, and setting it equal to zero, (4.18) can be written in matrix form as:

$$\mathbf{0} = \frac{1}{2}\mathbf{K} - \mathbf{V} + A\mathbf{VL} + B\mathbf{LV} + C\mathbf{KVK} - C\eta\mathbf{K} + D\mathbf{dVS},$$

where $\mathbf{L}, \mathbf{d}, \mathbf{K}$, and $\mathbf{S}$ are $\eta \times \eta$ matrices given by:

$$\mathbf{L} = \begin{bmatrix} 0 & 1 & \dots & 1 \\ 1 & \ddots & & \vdots \\ \vdots & & \ddots & 1 \\ 1 & \dots & 1 & 0 \end{bmatrix}, \mathbf{d} = \begin{bmatrix} d_{11} & \dots & d_{1\eta} \\ \vdots & \ddots & \vdots \\ d_{\eta 1} & \dots & d_{\eta\eta} \end{bmatrix}, \mathbf{K} = \begin{bmatrix} 1 & \dots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \dots & 1 \end{bmatrix},$$

and

$$\mathbf{S} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 1 \\ 1 & \ddots & \ddots & & & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & & & \ddots & \ddots & 1 \\ 1 & 0 & \dots & 0 & 1 & 0 \end{bmatrix}.$$

After some algebra, this becomes

$$\begin{aligned} 2u_o C\eta\mathbf{K} - \frac{1}{2}\mathbf{K} &= [(B+C)\mathbf{L} + (C-1)\mathbf{I}]\mathbf{V} + [C\mathbf{L} + (A+C)\mathbf{I}]\mathbf{V}\mathbf{L} \\ &\quad + D\mathbf{d}\mathbf{V}\mathbf{S}, \end{aligned}$$

or more simply

$$\mathbf{E} = \mathbf{\Gamma}\mathbf{V} + \mathbf{\Delta}\mathbf{V}\mathbf{L} + D\mathbf{d}\mathbf{V}\mathbf{S}, \tag{4.20}$$

where $\mathbf{E} = 2u_o(C\eta - \frac{1}{2})\mathbf{K}$, $\mathbf{\Delta} = C\mathbf{L} + (A+C)\mathbf{I}$, and $\mathbf{\Gamma} = (B+C)\mathbf{L} + (C-1)\mathbf{I}$.

The matrix $\mathbf{\Delta}$ is nonsingular, and its inverse is given by

$$\mathbf{\Delta}^{-1} = \begin{bmatrix} \sigma & -\psi & \dots & -\psi \\ -\psi & \ddots & & \vdots \\ \vdots & & \ddots & -\psi \\ -\psi & \dots & -\psi & \sigma \end{bmatrix},$$

where $\sigma = \frac{(\eta-1)A^{\eta-2}C + A^{\eta-1}}{A^\eta + \eta A^{\eta-1}C}$ and $\psi = \frac{A^{\eta-2}C}{A^\eta + \eta A^{\eta-1}C}$.

Equation (4.20) can then be written as

$$\mathbf{\Delta}^{-1}\mathbf{E} = \mathbf{\Delta}^{-1}\mathbf{\Gamma}\mathbf{V} \cdot \mathbf{V}\mathbf{L} + D\mathbf{\Delta}^{-1}\mathbf{d}\mathbf{V}\mathbf{S}. \tag{4.21}$$

Now define a matrix $\mathbf{P} = \mathbf{\Delta}^{-1}\mathbf{\Gamma}$ as

$$\mathbf{P} = \begin{bmatrix} \sigma(C-1) - \psi(\eta-1)(B+C) & & \sigma(B+C) - \psi(C-1) - \psi(\eta-2)(B+C) \\ & \ddots & \\ \sigma(B+C) - \psi(C-1) - \psi(\eta-2)(B+C) & & \sigma(C-1) - \psi(\eta-1)(B+C) \end{bmatrix}.$$

If the off-diagonal elements are set to zero, so that

$$\frac{\sigma}{\psi} = \frac{(\eta-2)(B+C)+(C-1)}{B+C}, \tag{4.22}$$

then $\mathbf{P}=[\sigma(C-1)-\psi(\eta-1)(B+C)]\mathbf{I}=Q\mathbf{I}$. Equating the expression for σ in $\mathbf{\Delta}^{-1}$ with the expression for σ in (4.22) yields the following:

$$\frac{AB}{C} + A + B = -1. \tag{4.23}$$

Using this relationship gives $Q=1+\frac{B}{C}$.

Equation (4.21) now becomes

$$\begin{aligned} \mathbf{\Delta}^{-1}\mathbf{E} &= Q\mathbf{V} + \mathbf{V}\mathbf{L} + D\mathbf{\Delta}^{-1}\mathbf{d}\mathbf{V}\mathbf{S} \\ &= \mathbf{V}(Q\mathbf{I}+\mathbf{L}) + D\mathbf{\Delta}^{-1}\mathbf{d}\mathbf{V}\mathbf{S}, \end{aligned}$$

or since $Q\mathbf{I}+\mathbf{L}$ is nonsingular,

$$\mathbf{E}(Q\mathbf{I}+\mathbf{L})^{-1} = \mathbf{\Delta}\mathbf{V} + \mathbf{d}\mathbf{V}\mathbf{S}(Q\mathbf{I}+\mathbf{L})^{-1}D.$$

Assuming the distance matrix $\mathbf{d}$ is nonsingular, this becomes

$$\mathbf{d}^{-1}\mathbf{E}(Q\mathbf{I}+\mathbf{L})^{-1} = \mathbf{d}^{-1}\mathbf{\Delta}\mathbf{V} + \mathbf{V}\mathbf{S}(Q\mathbf{I}+\mathbf{L})^{-1}D. \tag{4.24}$$

Define the following quantities:

$$\begin{aligned} \tilde{\mathbf{A}} &= \mathbf{d}^{-1}\mathbf{\Delta} \\ \tilde{\mathbf{B}} &= \mathbf{S}(Q\mathbf{I}+\mathbf{L})^{-1}D \\ \tilde{\mathbf{C}} &= \mathbf{d}^{-1}\mathbf{E}(Q\mathbf{I}+\mathbf{L})^{-1}. \end{aligned}$$

So (4.24) becomes

$$\tilde{\mathbf{C}} = \tilde{\mathbf{A}}\mathbf{V} + \mathbf{V}\tilde{\mathbf{B}}. \tag{4.25}$$

From Bellman [3], if the conditions of the following theorem are satisfied, the solution to (4.25) is given by

$$\mathbf{V} = -\int_0^\infty e^{\tilde{\mathbf{A}}t}\tilde{\mathbf{C}}e^{\tilde{\mathbf{B}}t}dt. \tag{4.26}$$

Theorem 1 *A necessary and sufficient condition that (4.25) have a solution for all* $\tilde{\mathbf{C}}$ *is that* $\lambda_i + \mu_j \neq 0$ *where* λ_i *are the characteristic roots of* $\tilde{\mathbf{A}}$ *and* μ_j *the characteristic roots of* $\tilde{\mathbf{B}}$.

The characteristic roots of the matrices $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$ must be calculated, and the values of the scalars A, B, C and D adjusted so as to satisfy the relationship in (4.25), and such that the characteristic roots meet the requirements of the theorem. In order to do this, the distance matrix $\mathbf{d}$ must be defined. This implies that the scalars are dependent on the values of the elements of $\mathbf{d}$, which ultimately depend on the arrival times of the attackers. This suggests that the scalar values are situation dependent, and must be calculated at run time.

Chapter 5

Numerical Results

In this chapter, the results of applying Algorithms 1, 2, and 3, and the neural network to several attack scenarios, in which the arrival times and arrival angles of the attackers are known a priori, are presented. The performance of the neural network under a deterministic system is discussed. In the deterministic formulation, the tracking methods of all the attackers are assumed to be centroiding (CEN). In the Bayesian formulation, the distribution describing the tracking method of an attacker is given by

$$\begin{aligned} Pr(\alpha = LE) &= .3 \\ Pr(\alpha = CEN) &= .7. \end{aligned}$$

It was assumed that the defensive resources are inexhaustible. It was also assumed that no updates of the attackers states were provided.

A total of nine attack scenarios were considered. The scenarios include situations in which one of the decoy types has a higher probability of success against a half or more of the attackers than the other type, and also include arrival angles for which the decoy type with the highest probability of survival depends on the tracking method of the attacker. The specifics of the scenarios are given in Table 5.1. The ninth attack scenario was included to test the neural network. It differs from the eighth in the spacing of the arrival times, and results in an increased number of decision instants, which in turn results in a larger neuron matrix. The probability distribution functions describing the success of the countermeasures are given in Figures 5.1, 5.2, and 5.3.

In all nine scenarios, the model parameters took the following values:

$$r_A = 16 \text{ miles}$$

$$v_A = .2814 \text{ miles/sec}$$

$$\Delta = 1 \text{ sec}$$

$$\Delta_D = 20 \text{ secs}$$

$$\Delta_B = 2 \text{ secs}$$

$$\mu_D = 20 \text{ secs}$$

$$r_B = 2.1 \text{ miles}$$

$$r_D = 5 \text{ miles.}$$

Tables 5.2-5.10 give the optimal allocation schemes determined by Algorithms 1, 2, and 3 for the specific attack scenarios. In each scheme, the decision

instants at which the "do nothing" control was the only feasible control, are not shown.

The optimal allocation scheme under the deterministic formulation in Table 1, is interpreted as follows: at the first decision instant, (time -20 secs) deploy one decoy of type R. At the next decision instant at which there are controls other than just Z feasible, (time 0 secs) deploy another decoy of type R, and so on. At time 40 secs, it is feasible to deploy either type of decoy, or "do nothing". The "do nothing" control is applied at this decision instant because the attacker has reached the range r_D, and since deploying either decoy has a zero probability of causing the attacker to transition from the tracking state into the not-tracking state, the target opts to "do nothing" rather than waste a decoy. At time 50 secs, the attacker is in range of the hardkill system, and this resource is allocated every 2 secs until time Ω=54 secs. The allocation schemes in the minimax and Bayesian formulations, and in Tables 5.3-5.10 are interpreted in a similar manner.

As can be seen, the optimal allocation schemes are very dependent on the specifics of the attack scenarios, i.e. the arrival times and the arrival angles of the attackers. For a specific scenario, the optimal allocation scheme also depends on how much information is known about the tracking methods of the attackers. As expected, the probability of target survival drops in both the minimax formulation and the Bayesian formulation, particularly when the number of attackers is greater than one.

The neural network was applied to the attack scenarios given in Table 5.1. The network was run 87 times for each scenario. The resulting allocation

schemes were compared to the allocation schemes known to be optimal as determined by Algorithm 1. The performance of the network is presented in Table 5.11.

Several observations were noted about the behavior of the network. The network was very sensitive to the values of the scalars A, B, C, and D in (4.11), and to the initial values of the neurons. Small changes in the value of any one of these parameters led to entirely different output matrices, some of which had elements with values between 0 and 1. Through trial and error, the following values of A, B, C, and D, and the external input currents resulted in the elements of the output matrices taking values of either 0 or 1, and were the values used in each run of the network:

$$A = 400 \quad B = 800 \quad C = 200 \quad D = 500$$

$$I_{Xi} = C(K + 5).$$

In each of the runs, the network started from a different random initial state. From Wilson and Pawley [4], the initial condition of each neuron was calculated using the following:

$$\begin{aligned} V_{Xi} &= \frac{1}{2}(1 + \tanh(\frac{x_{init}}{u_o})) \\ x_{init} &= -\frac{u_o}{2}\ln(K - 1) + \delta. \end{aligned}$$

where u_o=.02, and δ is a uniformly distributed random variable on the interval $[-.1x_{init}, .1x_{init}]$.

Whether or not the network converged to a valid allocation scheme depended solely on the initial values of the neurons. The invalid schemes were characterized by column errors, in which more than one 1 appeared in a column, or no

1's appeared. This is interpreted as the scheme allocating more than one resource simultaneously at the decision instant corresponding to the column, or allocating no resource at the decision instant.

In the majority of the invalid schemes, or the schemes that were valid but not optimal, segments of the schemes were optimal, but the entire scheme was either not valid or globally optimal. For instance, several schemes gave the optimal decoy deploying sequence, but an invalid or suboptimal hardkill sequence, and vice versa.

It is observed, that when the number of attackers was greater than one, the percentage of valid schemes decreased to zero. In the majority of these situations, the network resulted in schemes in which the hardkill sequences were optimal, but at the decision instants at which decoys were feasible, both decoy types were allocated.

These results do not show to what degree the size of the network affects its performance.

One last note, when all of the neurons in the network were allowed to vary, the schemes the network converged to were never valid, and exhibited almost no local optimality. When the output of certain neurons were fixed, as described in Section 4.1.1., the network began to yield valid and sometimes optimal allocation schemes, and when not valid the schemes often had optimal segments.

Attack situation	(θ_i, τ_i)	Ω
1	(60°,0sec)	54sec
2	(180°,0)	54
3	(30°,0)	54
4	(60°,0),(120°,2)	54
5	(30°,0),(180°,4)	58
6	(110°,0),(180°,4)	58
7	(20°,0),(50°,2),(50°,4)	56
8	(30°,0),(60°,2),(140°,4)	56
9	(30°,0),(60°,4),(140°,6)	62

Table 5.1: Attack scenarios.

Formulation	$\vec{u}_{opt}$	Probability of target survival
Deterministic	$RRRZG_1G_1$	.91
Minimax	$RRRZG_1G_1$	.89
Bayesian	$RRRZG_1G_1$	.90

Table 5.2: Attack scenario #1.

Formulation	$\vec{u}_{opt}$	Probability of target survival
Deterministic	$LLLZG_1G_1$	.80
Minimax	$RRRZG_1G_1$	.19
Bayesian	$LLLZG_1G_1$	.75

Table 5.3: Attack scenario #2.

Formulation	$\vec{u}_{opt}$	Probability of target survival
Deterministic	$RRRZG_1G_1$	.94
Minimax	$RRRZG_1G_1$	.93
Bayesian	$RRRZG_1G_1$	.93

Table 5.4: Attack scenario #3.

Formulation	$\vec{u}_{opt}$	Probability of target survival
Deterministic	$LRRZG_1G_2$	.56
Minimax	$RLRZG_1G_2$	.46
Bayesian	$LRRZG_1G_2$	.51

Table 5.5: Attack scenario #4.

Formulation	$\vec{u}_{opt}$	Probability of target survival
Deterministic	$LLLZG_1G_1G_2G_2$	.59
Minimax	$RRRZG_1G_1G_2G_2$	.18
Bayesian	$LLLZG_1G_1G_2G_2$	.46

Table 5.6: Attack scenario #5.

Formulation	$\vec{u}_{opt}$	Probability of target survival
Deterministic	$LLLZG_1G_1G_2G_2$	.51
Minimax	$RRRZG_1G_1G_2G_2$	.04
Bayesian	$LLLZG_1G_1G_2G_2$	.40

Table 5.7: Attack scenario #6.

Formulation	$\vec{u}_{opt}$	Probability of target survival
Deterministic	$RRRZG_1G_2G_3$	.70
Minimax	$RRRZG_1G_2G_3$	.63
Bayesian	$RRRZG_1G_2G_3$	.67

Table 5.8: Attack scenario #7.

Formulation	$\vec{u}_{opt}$	Probability of target survival
Deterministic	$RRRZG_1G_2G_3$	.56
Minimax	$RRRZG_1G_2G_3$	.46
Bayesian	$RRRZG_1G_2G_3$	.52

Table 5.9: Attack scenario #8.

Formulation	$\vec{u}_{opt}$	Probability of target survival
Deterministic	$RRRZG_1G_1G_2G_2G_3G_3$	.56
Minimax	$RRRZG_1G_1G_2G_2G_3G_3$	.5
Bayesian	$RRRZG_1G_1G_2G_2G_3G_3$	.57

Table 5.10: Attack scenario #9.

Attack situation	Network size	Valid schemes found	Optimal schemes found
1	13×5	21	5
2	13×5	72	30
3	13×5	9	1
4	13×5	0	0
5	17×7	0	0
6	17×7	0	0
7	15×6	0	0
8	15×6	0	0
9	23×9	0	0

Table 5.11: Performance of the neural net.

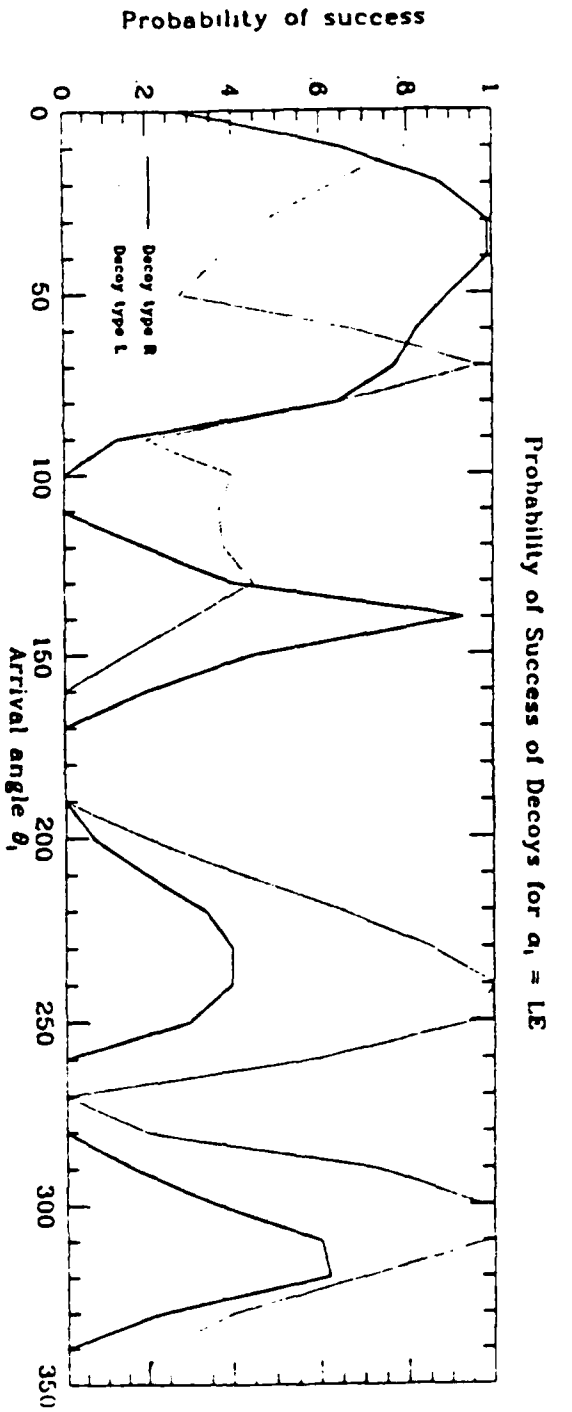

Figure 5.1: Probability of success of decoys with $\alpha = LE$

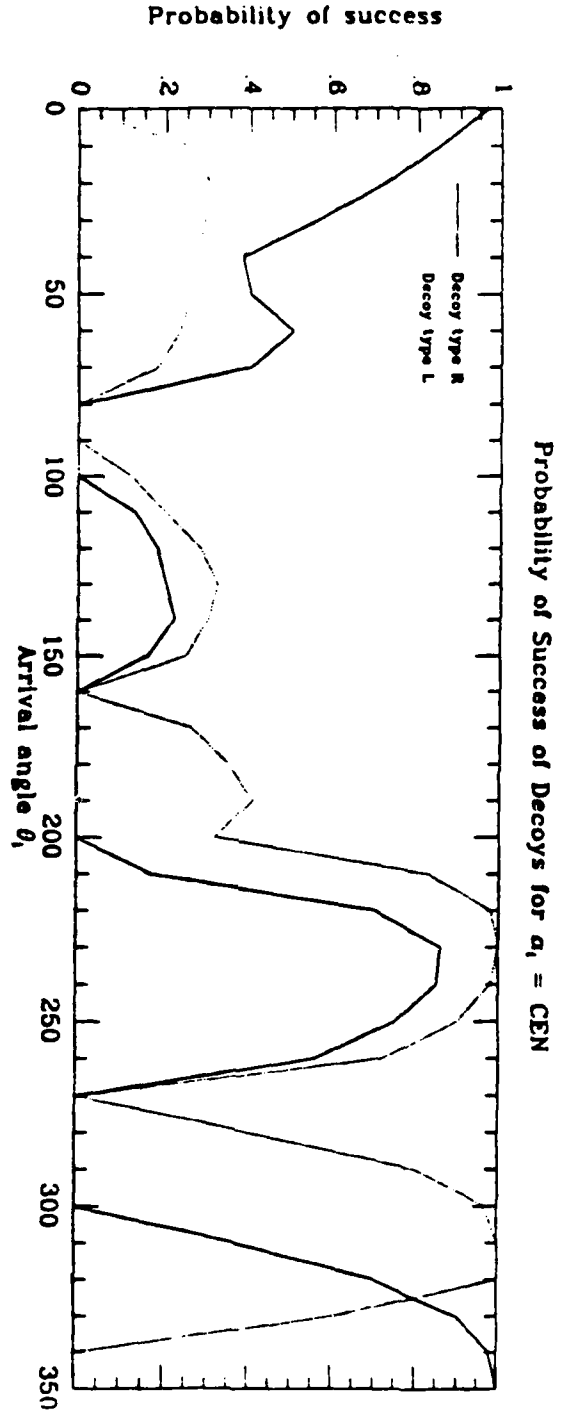

Figure 5.2: Probability of success of decoys with $\alpha = CEN$

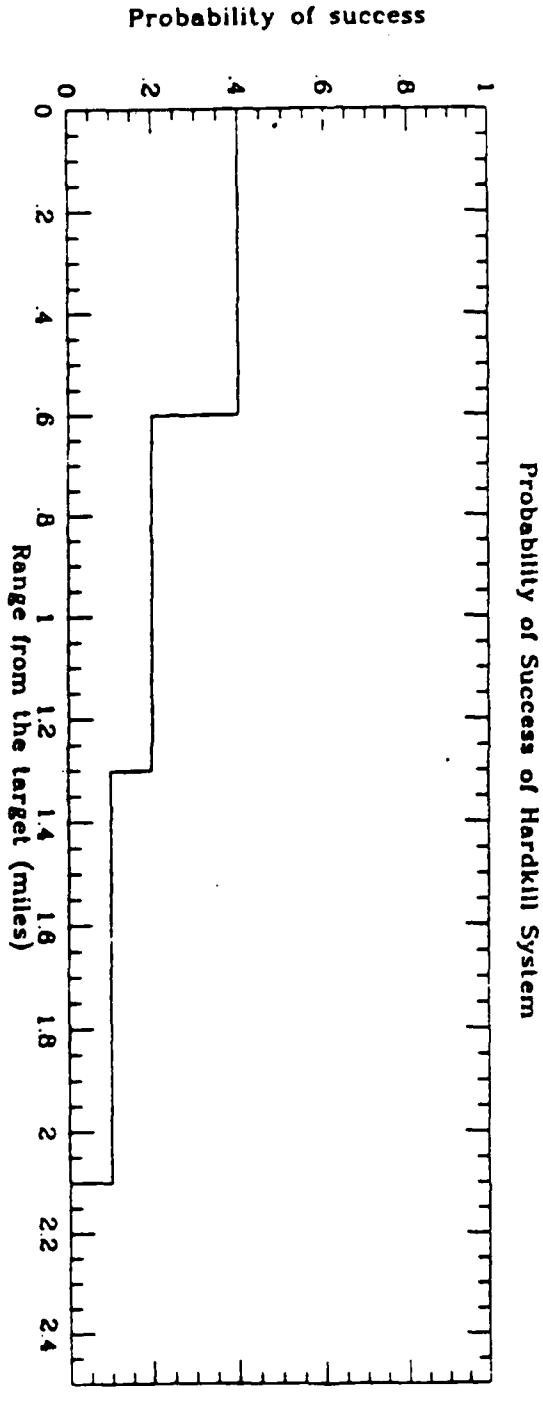

Figure 5.3: Probability of success of hardkill system

Conclusion

Algorithms 1, 2, and 3 are easily implementable, and require less computation to determine the optimal allocation schemes than their exhaustive search counterparts. The resulting schemes are totally dependent on the attack scenario, and the amount of a priori information available about the tracking methods of the attackers.

The model becomes more complex as the assumptions are removed. Relaxing the assumption that the target is stationary greatly affects the probability of target survival by making it a continuous function of time. As explained in the Introduction, one parameter the probability of success of a countermeasure depends on, and thus the probability of target survival, is the physical location of the attacker relative to the target. Permitting the target to move therefore results in time-varying $P^i_{u_k}$'s. The values of the $P^i_{u_k}$'s cannot be predicted unless

the trajectory of the target is known.

Allowing the target to move can be considered a countermeasure. If the angle of an attacker is such that the probabilities of success of the decoys or the hardkill system are very low, the target may move and turn, so that the attacker ends up in a position relative to the target for which the probability of success at least one of the countermeasures is higher. However by doing this, the target may have made itself more vulnerable to other attackers, present or future; and may also have excluded certain countermeasures from being currently feasible. If the arrival times and arrival angles of the attackers are known a priori, the target can plan its trajectory and determine an allocation scheme so that its probability of survival is the highest.

Relaxing the assumption that there is only one attacker type complicates the model only if the attacker types are unknown, and there are countermeasures which are only affective on certain attacker types.

Also the issue of constrained resources was not addressed. Reformulating the model to allow a mobile target, and developing algorithms for determining the optimal allocation schemes in each of the three formulations, particularly when the arrival times are not known a priori, and there is a constrained number of resources, is a possible consideration for future research.

Although fixing the output of certain neurons drastically improved the performance of the network, the neural network does not appear to be a reliable means of determining optimal allocation schemes. In the case of a single attacker, the network converged to valid schemes in 39% of the runs, and to the optimal allocation scheme in 14% of the runs. In all cases of two or more at-

tackers, no valid allocation schemes were determined, however long segments of the schemes were frequently optimal. The errors in the invalid schemes were typically the allocation of both decoy types at a decision instant at which decoys were feasible, (column error). The attackers seem to be "competing" for the decoy that is least likely to succeed against him. This results in an invalid scheme.

However, the fact that the network does sometimes find the optimal allocation scheme, and that it tends to locally optimize, suggests simulated annealing as a way of using the neural network to find the optimal allocation schemes.

For the minimax and Bayesian formulations, and when arrival states of the attackers are not known a priori, it is not clear how to construct the network to find the optimal allocation schemes. In the Bayesian formulation and the unknown arrivals case, the cost function is a weighted sum of products. If one network is to be used, it is uncertain how to bound the cost function, and calculate a distance matrix that reflects the weights in the sum; or if several networks are to be used, it is uncertain how to link them together.

The method described in Section 4.2 was not used to choose the values of the scalars A, B, C, and D in (4.11). This method assumes a square neuron matrix, and the neuron matrix in the network developed in Chapter 4 was not square. However the analysis was helpful in giving a little more insight as to the behavior of the network. The same set of scalar values led to stable networks in the nine scenarios considered. This suggests that it is only required that relationships in the form of inequalities between the scalars be met. This seems plausible since for (4.25) to have a solution, Theorem 1 only requires that none

of the cross sums of the eigenvalues of the matrices $\tilde{\mathbf{A}}$ and $\tilde{\mathbf{B}}$ equal zero. The exact inequalities that must be satisfied, depend on the values of the elements of the distance matrix $\mathbf{d}$, which explains why certain values of the scalars result in an unstable network.

Appendix A

Energy Function Describing Neural Network

The derivation of the expression in (4.11) from the expression in (4.12) using the equations in (4.13) and (4.14) is given. The equations in (4.13) and (4.14) are repeated below for convenience:

$$\begin{aligned} I_{Xi} &= CK \\ T_{Xi,Yj} &= -A\delta_{XY}(1-\delta_{ij}) - B\delta_{ij}(1-\delta_{XY}) - C \\ &\quad -Dd_{XY}(\delta_{j,i+1} + \delta_{j,i-1}) \\ \delta_{ij} &= \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases} \end{aligned} \tag{A.1}$$

From (4.11),

$$E_1 = -\frac{1}{2}\sum_X \sum_Y \sum_i \sum_j T_{Xi,Yj} V_{Xi} V_{Yj} - \sum_X \sum_i V_{Xi} I_{Xi}. \tag{A.2}$$

Substituting (A.1) into (A.2) gives (4.11),

$$
\begin{aligned}
= & -\frac{1}{2}\sum_X\sum_Y\sum_i\sum_j[-A\delta_{XY}(1-\delta_{ij})-B\delta_{ij}(1-d_{XY}) \\
& -C-Dd_{XY}(\delta_{j,i+1}+\delta_{j,i-1})]V_{Xi}V_{Yj}-\sum_X\sum_i V_{Xi}I_{Xi} \\
= & \frac{1}{2}[A\sum_X\sum_i\sum_{j\neq i}V_{Xi}V_{Xj}+B\sum_X\sum_{Y\neq X}\sum_i V_{Xi}V_{Yi} \\
& +C\sum_X\sum_Y\sum_i\sum_j V_{Xi}V_{Yj}+D\sum_X\sum_{Y\neq X}\sum_i d_{XY}V_{Xi}(V_{Y,i+1}+V_{Y,i-1})] \\
& -CK\sum_X\sum_i V_{Xi} \\
= & \frac{A}{2}\sum_X\sum_i\sum_{j\neq i}V_{Xi}V_{Xj}+\frac{B}{2}\sum_X\sum_{Y\neq X}\sum_i V_{Xi}V_{Yi} \\
& +\frac{C}{2}(\sum_X\sum_i V_{Xi}-K)^2-\frac{CK^2}{2} \\
& +\frac{D}{2}\sum_X\sum_{Y\neq X}\sum_i d_{XY}V_{Xi}(V_{Y,i+1}+V_{Y,i-1}).
\end{aligned}
$$

The equation describing the dynamics of each neuron given in (4.15) is derived from (4.3) using the equations (A.1).

$$
\begin{aligned}
\frac{dx_{Xi}}{dt} = & -\frac{x_{Xi}}{RC}+\sum_Y\sum_j T_{Xi,Yj}V_{Yj}+I_{Xi} \\
= & -\frac{x_{Xi}}{RC}+\sum_Y\sum_j[-A\delta_{XY}(1-\delta_{ij})-B\delta_{ij}(1-d_{XY})-C \\
& -Dd_{XY}(\delta_{j,i+1}+\delta_{j,i-1})]V_{Yj}+CK \\
= & -\frac{x_{Xi}}{RC}-A\sum_{j\neq i}V_{Xj}-B\sum_{Y\neq X}V_{Yi}-C\sum_Y\sum_j V_{Yj}+CK \\
& -D\sum_Y d_{XY}(V_{Y,i+1}+V_{Y,i-1}) \\
= & -\frac{x_{Xi}}{RC}-A\sum_{j\neq i}V_{Xj}-B\sum_{Y\neq X}V_{Yi}-C(\sum_Y\sum_j V_{Yj}-K) \\
& -D\sum_Y d_{XY}(V_{Y,i+1}+V_{Y,i-1}).
\end{aligned}
$$

References

[1] J. Hopfield and D. Tank, "Neural computation of decisions in optimization problems," *Biological Cybernetics*, vol. 52, pp. 141–152, 1985.

[2] J. Hopfield, "Neuron with graded response have collective computational properties like those of two state neurons," *Proc. Natl. Acad. Sci. USA*, vol. 81, pp. 3088–3092, 1984.

[3] R. Bellman, *Introduction to Matrix Analysis.* New York: McGraw-Hill Book Company, 1970.

[4] G. Wilson and G. Pawley, "On the stability of the travelling salesman problem algorithm of hopfield and tank," *Biological Cybernetics*, vol. 58, pp. 63–70, 1988.

[5] H. Szu, "Fast tsp algorithm based on binary neuron output and analog neuron input using the zero-diagonal interconnect matrix and necessary and sufficient constraints of the permutation matrix," Technical Report, Naval Research Laboratory, 1989.

[6] C. W. Helstrom, *Probability and Stochastic Processes for Engineers.* New York: Macmillan Publishing Company, 1984.

[7] D. G. Luenberger, *Linear and Nonlinear Programming.* New York: Addison-Wesley Publishing Company, 1984.