GAME THEORETIC TARGET ASSIGNMENT STRATEGIES IN COMPETITIVE MULTI-TEAM SYSTEMS

by

David G. Galati

BS, University of Pittsburgh, 2000

MS, University of Pittsburgh, 2002

Submitted to the Graduate Faculty of

School of Engineering in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2004

UNIVERSITY OF PITTSBURGH

SCHOOL OF ENGINEERING

This dissertation was presented

by

David G. Galati

It was defended on

December 6th 2004

and approved by

Dr. James Antaki, Adjunct Associate Professor, Bioengineering Department

Dr. J. Robert Boston, Professor, Department of Electrical Engineering

Dr. Luis F. Chaparro, Associate Professor, Department of Electrical Engineering

Dr. Ching-Chung Li, Professor, Department of Electrical Engineering

Dissertation Director: Dr. Marwan A. Simaan, Bell of Pennsylvania/Bell Atlantic Professor, Department of Electrical Engineering

GAME THEORETIC TARGET ASSIGNMENT STRATEGIES
IN COMPETITIVE MULTI-TEAM SYSTEMS

Dr. David G. Galati, PhD

University of Pittsburgh, 2004

The task of optimally assigning military ordinance to enemy targets, often termed the Weapon Target Assignment (WTA) problem, has become a major focus of modern military thought. Current formulations of this problem consider the enemy targets as either passive or entirely defensive. As a result, the assignment problem is solved purely as a one sided team optimization problem. In practice, however, especially in environments characterized by the presence of an intelligent adversary, this one sided optimization approach has very limited use. The presence of an adversary often necessitates incorporating its intended actions in the process of solving the weapons assignment problem. In this dissertation, we formulate the weapon target assignment problem in the presence of an intelligent adversary within the framework of game theory. We consider two teams of opposing units simultaneously targeting each other and examine several possible game theoretic solutions of this problem. An issue that arises when searching for any solution is the dimensionality of the search space which quickly becomes overwhelming even for simple problems with a small number of units on each side. To solve this scalability issue, we present a novel algorithm called Unit Level Team Resource Allocation (ULTRA), which is capable of generating approximate solutions by searching within appropriate subspaces of the search space. We evaluate the performance of this algorithm on several realistic simulation scenarios. We also show that this algorithm can be effectively implemented in real-time as an automatic target assigning controller in a dynamic multi-stage problem involving two teams with large number of units in conflict.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

PREFACE

A large body of work, such as this dissertation, is simply not possible without the help, support and encouragement of a number of individuals. First and foremost, I would like to thank my advisor, Marwan Simaan, for his patience, advice and guidance. Without his mentorship, none of this would have been possible. I would also like to thank Jim Antaki, Bob Boston, Luis Chaparro and C.C. Li for taking the time and effort to serve on my committee. Reviewing a 150 page document is by no means trivial, and their comments, questions and criticisms proved an invaluable tool throughout the research process. Special thanks to Yong Liu, for her ideas and insights throughout our project as well as her company in the lab. I'd like to thank my family, Rachel, Becky, Steve, Dan, Ben and especially my parents, Larry and Carol, for their eternal devotion and support throughout all of my endeavors. Lastly, thanks to my cousin Mike for providing a necessary distraction throughout the writing process.

This research was sponsored in part by the Defense Advanced Research Projects Agency (DARPA) under contract number F33615-01-C-3151. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the DARPA, or the U.S. Government.

1.0 INTRODUCTION

Today's modern military systems rely heavily on the use of smart and highly efficient weapons. To produce these weapons unprecedented levels of funds must be expended. Due to the expense inherent in producing these weapons, their efficient use in a military engagement has become one of the driving forces in modern military thought. The weapon target assignment (WTA) problem has consequently received a great deal of attention in recent years [1-11]. A form of the general non-linear assignment problem, the WTA problem is typically formulated as a single team of military units which must be optimally assigned enemy target units in such a way as to optimize some objective function related to the effectiveness of the assignment. In this dissertation, we will focus on a more general formulation of the weapon target assignment problem. This formulation considers a situation in which two or more teams of military units are involved, and when units in each team are simultaneously targeting units in the remaining teams. We will examine the single-team weapon target (ST-WTA) allocation problem.

In the most general formulation of the WTA problem, a team of "weapons" must each be assigned to one of a series of targets in such a manner as to optimize some objective function. Typically, this objective function may represent a measure of destruction of the targets, although other measures, such as the cost of each weapon, may also be considered. It should be noted that the term "weapons" need not be confined to missiles or bombs, although this is the most common interpretation. A "weapon" may represent a higher level of military ordinance such as a tank, plane, destroyer or it may represent even larger groups such as entire battalions or fleets. In

fact a “weapon” need not have any military connotation whatsoever. It may be thought of as any entity related to each of the given targets by means of some non-decreasing probability function. Furthermore, while in a military situation the non-decreasing probability function relationship between weapon and target is typically defined as a probability of kill or a probability of damage, it may be thought of more generally as a probability of task achievement. However, to maintain clarity, in this dissertation the military convention will be utilized. A “team” will be defined as a group of entities having a common goal. These entities will be denoted as “units”. A unit can best be thought of in terms of a military weapons platform. Thus, depending on the application, a unit may be capable of deploying a number of weapons, or the unit itself may in fact be a weapon.

The WTA problem comes in two flavors, the static weapon target assignment (SWTA) problem and the dynamic weapon target assignment (DWTA) problem. The SWTA and the DWTA problem formulations differ in that the STWA problem assumes that all weapons are allocated to targets at only one time, while the DTWA problem assumes that this occurs in several discrete “battle-steps.” As the SWTA problem is the simpler of the two, in order to illustrate the problem analytically, let us consider the case where a team (say team A) of N units is attacking another team (say team B) of M targets. Let $p_{i,j}$ represent the probability of the i^{th} unit in team A destroying the j^{th} target in team B. If $p_{i,j}$ is a Bayesian probability of kill, then the probability that the j^{th} target will survive an attack by the i^{th} unit is given by $q_{i,j} = 1 - p_{i,j}$. Further, if these probabilities are assumed to be independent, then the probability T_j that the j^{th} target will survive when attacked by more than one weapon, can be defined as the product of the

probabilities of survival $q_{i,j}$ of each of the units assigned to the given target, as described in the attrition model defined by Manne [1].

$$T_j = \prod_{i=1}^{N} \left[q_{i,j}^{(u_{i,j})} \right] \tag{1.1}$$

Here $u_{i,j}$ represents the number of weapons the i^{th} unit in team A assigns to the j^{th} target in team B. Furthermore, the number of weapons carried on unit i is assumed to be limited by an upper bound W_i. This places the following restrictions upon (1.1):

$$\sum_{j=1}^{M} u_{i,j} \le W_i, \ \ \forall i = 1, 2, ..., N \tag{1.2a}$$

$$u_{i,k} \ge 0 \text{ and is an integer } \forall i = 1, 2, ..., N, \text{ and } j = 1, 2, ..., M \tag{1.2b}$$

Each target j is assumed to have some defined value to team A. Let this be c_j. Thus an objective function $J(u)$ can be contrived as the sum of the scaled probabilities of survival, $c_j T_j$, over all possible targets. Defining the vector u as follows:

$$\begin{aligned} u &= (u_1, u_2, \ldots, u_N)' \\ u_i &= (u_{i,1}, u_{i,2}, \ldots, u_{i,M})' \end{aligned} \tag{1.3}$$

yields the following maximization:

$$\max_u J(u) = \sum_{j=1}^{M} c_j (1 - T_j) \approx \max_u \sum_{j=1}^{M} -c_j T_j \tag{1.4}$$

Consequently, (1.4) can be combined with (1.1) to yield the following possible complete description of the constraint optimization problem associated with the SWTA:

$$\max_{u} J(u) = \sum_{k=1}^{M} -c_k \prod_{i=1}^{N} \left[q_{i,j}^{u_{i,j}} \right] \quad \text{subject to:}$$

$$\sum_{j=1}^{M} u_{i,j} \le W_i, \ \forall i = 1,2,...,N \quad \text{and:} \tag{1.5}$$

$$u_{i,j} \ge 0 \text{ and is an integer } \forall i = 1,2,...,N, \text{ and } j = 1,2,...,M$$

It should be noted that (1.5) represents only a single possible objective function for SWTA problem. In general T_j may be any monotonically decreasing function of u, and c_j may also be a function of T_j. Other terms may also be included in the objective function, such as a penalty for the cost of every weapon launched.

While the SWTA has not been completely solved in general, a great deal of work has been done on generating possible solutions for special cases or approximate solutions for the general case. One common approximate solution is to relax the constraints on the problem by removing (1.2b). As non-integer solutions are then permitted, (1.5) can be solved exactly using non-linear programming techniques [3]. Various branch and bound techniques can then be used to reinstate (1.2b). Many heuristic approaches in addition to neural networks [4] and genetic approaches [5] are also employed. If all attacking units are assumed to have uniform (homogeneous) weapons (SWTA-U), a technique known as maximal marginal return has been proven to quickly arrive at an optimal solution [6]. In this technique, the attacking team is first assumed to consist of only a single unit. Once this unit has been optimally assigned to a target, a second unit is added and assigned to the target which yields the largest objective function increase. This process continues until all units have been assigned targets. Another variation of this technique, the SWTA-U problem is solved by continually swapping targets for units that increase the overall objective function [7]. Other more complicated algorithms have been developed for the DTWA

problem [8, 9]. Several more extensive literature surveys are available on the subject of the WTA problem [9-11] and the more general non-linear assignment problem [12].

1.1 MOTIVATION FOR THIS DISSERTATION

So far, the target assignment problem has been formulated only in terms of a single team of units with a given set of targets. This formulation is useful in cases where the adversary is assumed to be passive and not reacting to the actions of the attacking team. In most military combat situations, however, the adversary in reality could be active and intelligent. Thus, it is important to consider an extension of the WTA problem to cases where there are at least two opposing teams, each having some offensive and/or defensive capabilities. In this dissertation, we will refer to such problems as multi-team weapon target assignment problems (MT-WTA). The attrition model in these types of problems, and therefore the objective function of each team, depends on the target assignment strategies of all teams engaged in the battle. Consequently, the process of optimizing the target assignment strategy employed by one team must take into account the possible target assignments of all other teams. The SWTA techniques discussed earlier are only capable of dealing with a single team having a single objective function. As a result, approaches other than standard optimization methods are required to solve the MT-WTA problem.

To illustrate why such a method is required, consider an example of a MT-DWTA. A "Blue" team of N_B units is engaged in a multi-step combat against a "Red" team of N_R units. To generate a dynamic formulation of this problem it is necessary to separate the battle into

steps. If an attrition model similar to (1.4) is used, then the probability of a given unit destroying a target on the enemy team at a given battle step can be formulated as a function of the probability that the given unit survived the previous battle step in addition to the defined probability of kill. One possible attrition model for this problem is obtained from the following modification of (1.1):

$$\begin{aligned} B_i(k) &= B_i(k-1)\prod_{j=1}^{N_R}\left[\left(q_{j,i}^R(k)R_j(k-1)\right)^{u_{j,i}^R(k)}\right] \\ R_j(k) &= R_j(k-1)\prod_{i=1}^{N_B}\left[\left(q_{i,j}^B(k)B_i(k-1)\right)^{u_{i,j}^B(k)}\right] \end{aligned} \tag{1.6}$$

where $B_i(k)$ and $R_j(k)$ are the probabilities that the i^{th} and j^{th} blue and red units survive at the end of the k^{th} battle step. The target assignment strategies $u_{i,j}^B(k)$ and $u_{j,i}^R(k)$ for the Blue and Red teams are also assumed to vary at each battle step k. Consequently, this added variability necessitates further restrictions on the target assignment strategies. A unit may only launch a limited number of weapons per battle-step, and each unit has a finite number of weapons to launch over the course of the entire battle. These two conditions results in the following three constraints:

$$\begin{aligned} &\sum_{k=1}^{K}\sum_{j=1}^{N_R}u_{i,j}^B(k) \le W_i^{\mathrm{B}}, \ \ \forall i = 1,2,...,N_B \\ &\sum_{k=1}^{K}\sum_{i=1}^{N_B}u_{j,i}^R(k) \le W_j^{\mathrm{R}}, \ \ \forall j = 1,2,...,N_R \end{aligned} \tag{1.7a}$$

$$\begin{aligned} &\sum_{j=1}^{N_R}u_{i,j}^B(k) \le w_i^{\mathrm{B}}, \ \ \forall i = 1,2,...,N_B \\ &\sum_{i=1}^{N_B}u_{j,i}^R(k) \le w_j^{\mathrm{R}}, \ \ \forall j = 1,2,...,N_R \end{aligned} \tag{1.7b}$$

$$\begin{aligned} &u_{i,j}^B(k) \ge 0 \text{ and is an integer } \forall i = 1,2,...,N_B, \text{ and } j = 1,2,...,N_R, \text{ and } k = 1,2,...,K \\ &u_{j,i}^R(k) \ge 0 \text{ and is an integer } \forall i = 1,2,...,N_B, \text{ and } j = 1,2,...,N_R, \text{ and } k = 1,2,...,K \end{aligned} \tag{1.7c}$$

where w_i^B and w_j^R are the maximum number of weapons that may be launched by the i^{th} and j^{th} on the blue and red teams respectively, and K is the total number of battle steps. To further develop the MT-DWTA, we must consider the objective functions. One possible set of objectives is that each team desires to destroy as many as possible of its opponent's units and preserve as many as possible of its own. Assuming that c_{Bi}^B and c_{Rj}^B represent the value placed by the Blue team on the i^{th} Blue and j^{th} Red units respectively, and c_{Bi}^R and c_{Rj}^R represent the corresponding values for the Red team, then the objective functions can be expressed as follows:

$$\begin{aligned} J_B(k) &= \sum_{i=1}^{N_B} c_{Bi}^B B_i(k) - \sum_{j=1}^{N_R} c_{Ri}^B R_j(k) \\ J_R(k) &= -\sum_{i=1}^{N_B} c_{Bi}^R B_i(k) + \sum_{j=1}^{N_R} c_{Ri}^R R_j(k) \end{aligned} . \tag{1.8}$$

Because the state vectors $B(k)$ and $R(k)$ are dependent on the controls of each team it is clear that it is not possible to generate a team's optimum target assignment strategy without first knowing the target assignment strategy of the other team. Game theory provides widely accepted tools and solution concepts [13, 14] to deal with problems of this nature. In particular, the Nash strategy of game theory provides an equilibrium point among all of the objective functions which guarantees that no team will benefit from unilaterally modifying its given strategy. These strategies have recently received considerable attention, especially in the context of teaming and tasking in cooperative control systems in the presence of an adversary [15-17].

1.2 ORGANIZATION OF THIS DISSERTATION

This dissertation is comprised of 6 chapters. In the first chapter we will introduce the weapon target assignment problem and define the problem considered in this research. This chapter also contains the basic layout of the dissertation. In the second chapter we will review some of the basic concepts of game theory that are pertinent to the MT-WTA. In particular, we will define and discuss several game theoretic solution concepts including the Nash and Stackelberg strategies.

In the third chapter, we will consider a special case of the MT-WTA and we present a novel algorithm for determining the optimal target assignment strategy for one team when the target assignment strategy of the other team is known. This algorithm, which we refer to as: ULTRA for unit level team resource allocation algorithm, is capable of being implemented in a game theoretic framework. Issues such as convergence, computational complexity and accuracy are addressed and formulated into performance measures. These performance measures are then used to evaluate the ULTRA algorithm in several representative WTA problems.

In the fourth chapter, we plan to address the multi-team weapon target assignment problem. Here we will present the standard action - reaction Nash equilibrium search. We then combine this method of generating target assignment strategies with the ULTRA algorithm to quickly generate Nash equilibrium type target assignment strategies for the case of the two team WTA problem. We discuss the pros and cons of this method in various circumstances, including the cases of zero and multiple Nash equilibriums. As another point of interest, we comment on how

this method could be expanded to a more general MT-DWTA where three or more teams are engaged in combat. Lastly, we will evaluate the performance of this algorithm, comparing the MT-DWTA game theoretic approach to other target assignment strategy types.

In the fifth chapter, we will describe an implementation of the MT-DWTA model into a realistic battle field simulation as part of a mixed initiative hierarchical controller. Here we describe methods to incorporate distance and position into the MT-DWTA model including the distance discount factor. We also present methods for determining battle step duration, countermeasure deployment and command initiative. Lastly, in chapter six, we will summerize the main contributions of this dissertation and discuss possible future research directions.

2.0 INTRODUCTION TO GAME THEORY

In the previous section, we discussed a class of multi-team weapon target allocation problems that was not capable of beings solved by standard WTA optimization techniques. Standard WTA optimization techniques cannot incorporate the presence of an active intelligent adversarial team. The interdependence of objective functions in the MT-WTA necessitates the use of solution concepts from game theory. It is clear that a "good" strategy for any one team must take into account the possible target assignments of the opposing teams. It can also be argued that such a strategy should also assume that the other teams are capable of doing the same when calculating their target assignment strategies. The field of Game Theory was established to solve problems of this type. Generally speaking, Game Theory strives to define the characteristics of a "good" strategy and to derive methods of obtaining such strategies. As a result it provides a natural framework from which to solve the MT-DWTA.

To give the reader a greater understanding of game theory, we will provide a brief overview of the two most common game theoretic solution concepts, the Nash and Stackelberg equilibrium strategies. To illustrate the applicability of this theory, we solve a simple example of the MT-DWTA problem using the Nash strategies. We also discuss some possible obstacles that prevent these solution concepts from easily being applied to realistic, more complicated, problems.

The concept of game theory has it origins in the work of three individuals, Agustin Cournot, Joseph Bertrand, and Heinrich von Stackelberg [18-20]. In 1838 Cournot introduced the concept

of a “demand” function to relate profit and quantity produced. As a result, Cournot was able to derive models for monopolies, duopolies, and unlimited competition. Using an equilibrium concept, Cournot was able to show that competition between the producers would naturally promote lower prices than that observed in the case of a monopoly [18]. The work by Cournot was later reviewed in 1883 by Joseph Bertrand [19]. Bertrand revised Cournot’s work to pertain to the pricing of a product rather than the production level. He was then able to arrive at the same equilibrium concept. Stackelberg examined this issue again in 1934 [20]. Stackelberg objected to the work of Cournot and Bertrand because companies seldom select prices at the same time. In a true duopoly, one company sets the price for its products, and the second company is forced to react accordingly. He reasoned that in a duopoly, if the first producer has all of the information given to the second producer, the first producer would be able to predict the prices that the second chooses. As a result, by being first to announce a price, a producer has more control over its profits.

While Cournot, Bertrand, and Von Stackelberg introduced many of the concepts inherent in game theory, it was not known as Game Theory unil the work of John von Neumann in 1928 and Oskar Morgenstern in 1944 [21,22]. In his work, Von Neumann introduced the “mini-max” strategy or the strategy of minimum risk. In a mini-max strategy, a team selects the strategy with the best worst case scenario. Von Neumann also described the zero sum game, in which the sum of both team objective functions always sum to zero. He then proved that the mini-max strategy is an equilibrium strategy in a zero sum game. This work was later generalized by John Nash. Nash extended Neumann’s equilibrium concept to include non-zero sum games [13], a more general class of game in which the objective functions are not forced to sum to zero.

Today, nearly all the work done in Game Theory is dependent in some way upon the work of John Nash.

2.1 NASH EQUILIBRIUM STRATEGIES

The Nash equilibrium is one of the most important concepts in modern game theory. Simply stated, a set of strategies are defined as Nash if no decision-maker (or player) has an incentive to unilaterally alter its Nash strategy. To form a more mathematical expression, consider the general case of a two player game, between players A and B. Each player desires to maximize an given objective function $J_A(u_A, u_B)$ for player A and $J_B(u_A, u_B)$ for player B, both of which depend on the strategies, u_A and u_B selected by each player where $u_A \in U_A$ and $u_B \in U_B$. A set of given strategies, (u_A^N, u_B^N), are defined as Nash strategies if they satisfy the inequalities:

$$\begin{aligned} J_A(u_A^N, u_B^N) &\geq J_A(u_A, u_B^N) \quad \text{for all } u_A \in U_A \\ J_B(u_A^N, u_B^N) &\geq J_B(u_A^N, u_B) \quad \text{for all } u_B \in U_B \end{aligned} . \tag{2.1}$$

In the class of games where the strategy spaces are finite (cases where the strategies are discrete), the Nash solution is typically determined through the use of a game matrix. A game matrix is an array composed of columns and rows representing each possible strategy by each of the two sides. Each entry in the game matrix is a pair of real numbers that represent the values of the two players' objective functions for the given combination of strategies. Once a game matrix has been created, the optimal strategies of each player for every possible strategy for the opposing player are marked. These sets of optimal "reactions" are known as reaction sets. A

Nash solution can then be determined by searching for intersections of the reaction sets. To illustrate how a game matrix may be used to find a set of Nash strategies, consider the hypothetical game matrix shown in Table 2.1, in which players A and B must each choose from 4 possible strategies numbered 1 through 4.

Table 2.1 – Example of a Game Matrix

	$u_B = 1$	$u_B = 2$	$u_B = 3$	$u_B = 4$
$u_A = 1$	$J_A(1,1) = 5$ $J_B(1,1) = 15$	$J_A(1,2) = 6$ $J_B(1,2) = 4$	$J_A(1,3) = 14$ $J_B(1,3) = 1$	$J_A(1,4) = 0$ $J_B(1,4) = 20$
$u_A = 2$	$J_A(2,1) = 1$ $J_B(2,1) = 7$	$J_A(2,2) = 10$ $J_B(2,2) = 8$	$J_A(2,3) = 4$ $J_B(2,3) = 4$	$J_A(2,4) = 6$ $J_B(2,4) = 2$
$u_A = 3$	$J_A(3,1) = 4$ $J_B(3,1) = 6$	$J_A(3,2) = 2$ $J_B(3,2) = 7$	$J_A(3,3) = 20$ $J_B(3,3) = 3$	$J_A(3,4) = 8$ $J_B(3,4) = 3$
$u_A = 4$	$J_A(4,1) = 3$ $J_B(4,1) = 10$	$J_A(4,2) = 9$ $J_B(4,2) = 2$	$J_A(4,3) = 16$ $J_B(4,3) = 5$	$J_A(4,4) = 7$ $J_B(4,4) = 5$

The reaction sets can then be found by marking the optimal strategy for player A in each column and the optimal strategy for player B in each row. This operation is illustrated in Table 2.2. Finally, according to (2.1), a set of Nash strategies can be defined as the strategies representative of an entry in the game matrix which is marked as an optimal reaction for both player A and player B. In the case of table 2.2, such a set of strategies exists for $u_A = 2$ and $u_B = 2$ therefore $\left(u_A^N, u_B^N\right) = (2,2)$.

Table 2.2 – Reaction Sets of the Game Matrix

	$u_B = 1$	$u_B = 2$	$u_B = 3$	$u_B = 4$
$u_A = 1$	$J_A^*(1,1) = 5$ $J_B(1,1) = 15$	$J_A(1,2) = 6$ $J_B(1,2) = 4$	$J_A(1,3) = 14$ $J_B(1,3) = 1$	$J_A(1,4) = 0$ $J_B^*(1,4) = 20$
$u_A = 2$	$J_A(2,1) = 1$ $J_B(2,1) = 7$	$J_A^*(2,2) = 10$ $J_B^*(2,2) = 8$	$J_A(2,3) = 4$ $J_B(2,3) = 4$	$J_A(2,4) = 6$ $J_B(2,4) = 2$
$u_A = 3$	$J_A(3,1) = 4$ $J_B(3,1) = 6$	$J_A(3,2) = 2$ $J_B^*(3,2) = 7$	$J_A^*(3,3) = 20$ $J_B(3,3) = 3$	$J_A^*(3,4) = 8$ $J_B(3,4) = 3$
$u_A = 4$	$J_A(4,1) = 3$ $J_B^*(4,1) = 10$	$J_A(4,2) = 9$ $J_B(4,2) = 2$	$J_A(4,3) = 16$ $J_B(4,3) = 5$	$J_A(4,4) = 7$ $J_B(4,4) = 5$

This example illustrates one important property of the Nash strategies. It shows that a set of Nash strategies are not necessarily globally optimal. In fact, a Nash strategy is seldom a globally optimal strategy for either player. The objective function result for player A at the Nash equilibrium is 10, but there are higher values of the objective function (14, 16 and 20) at other strategy pairs. This discrepancy is even greater for player B, having an objective score at the Nash equilibrium that is 12 less than the most globally optimal value. However, in each instance, these scores depend on the other player selecting a strategy which they would unlikely consider.

It should also be noted that this hypothetical game has been carefully created. Slight changes in the games illustrate a problem in which Game Theory falters. This is because there is no guarantee that a game will have exactly one pair of Nash strategies. A game may have more than a single Nash equilibrium. Consider the game matrix shown in Table 2.3.

Table 2.3 – Example of a Game Matrix Having 2 Sets of Nash Strategies

	$u_B = 1$	$u_B = 2$	$u_B = 3$	$u_B = 4$
$u_A = 1$	$J_A(1,1) = 5$ $J_B(1,1) = 15$	$J_A(1,2) = 6$ $J_B(1,2) = 4$	$J_A(1,3) = 14$ $J_B(1,3) = 1$	$J_A(1,4) = 0$ $J_B^*(1,4) = 20$
$u_A = 2$	$J_A(2,1) = 1$ $J_B(2,1) = 7$	$J_A^*(2,2) = 10$ $J_B^*(2,2) = 8$	$J_A(2,3) = 4$ $J_B(2,3) = 4$	$J_A(2,4) = 6$ $J_B(2,4) = 2$
$u_A = 3$	$J_A(3,1) = 4$ $J_B(3,1) = 6$	$J_A(3,2) = 2$ $J_B^*(3,2) = 7$	$J_A^*(3,3) = 20$ $J_B(3,3) = 3$	$J_A^*(3,4) = 8$ $J_B(3,4) = 3$
$u_A = 4$	$J_A^*(4,1) = 8$ $J_B^*(4,1) = 10$	$J_A(4,2) = 9$ $J_B(4,2) = 2$	$J_A(4,3) = 16$ $J_B(4,3) = 5$	$J_A(4,4) = 7$ $J_B(4,4) = 5$

Though this game differs from that given in table 2.2 by only a single value, $J_A(4,1) = 3 \Rightarrow 8$, it possesses two distinct Nash equilibrium points. The justification for selecting a Nash strategy is that it is the optimal strategy for a player when the other player also selects a Nash strategy. Having two Nash strategies creates the possibility that each player will select its strategy based on a different equilibrium. In the game of Table 2.3 this corresponds to strategy pairs (4, 2) and (1, 7), each of which is considerably worse for at least one of the two players. In such cases Nash equilibrium strategies lose some credibility as "good" strategies. Nevertheless, it can still be argued that one's opponent is unlikely to select a strategy that is not Nash.

The other undesired possibility is that a game might not possesses a Nash equilibrium. Consider the game shown in Table 2.4.

Table 2.4 – Example of a Game Matrix Without a Set of Nash Strategies

	$u_B = 1$	$u_B = 2$	$u_B = 3$	$u_B = 4$
$u_A = 1$	$J_A^*(1,1) = 5$ $J_B(1,1) = 15$	$J_A(1,2) = 6$ $J_B(1,2) = 4$	$J_A(1,3) = 14$ $J_B(1,3) = 1$	$J_A(1,4) = 0$ $J_B^*(1,4) = 20$
$u_A = 2$	$J_A(2,1) = 1$ $J_B(2,1) = 7$	$J_A(2,2) = 8$ $J_B^*(2,2) = 8$	$J_A(2,3) = 4$ $J_B(2,3) = 4$	$J_A(2,4) = 6$ $J_B(2,4) = 2$
$u_A = 3$	$J_A(3,1) = 4$ $J_B(3,1) = 6$	$J_A(3,2) = 2$ $J_B^*(3,2) = 7$	$J_A^*(3,3) = 20$ $J_B(3,3) = 3$	$J_A^*(3,4) = 8$ $J_B(3,4) = 3$
$u_A = 4$	$J_A(4,1) = 3$ $J_B^*(4,1) = 10$	$J_A^*(4,2) = 9$ $J_B(4,2) = 2$	$J_A(4,3) = 16$ $J_B(4,3) = 5$	$J_A(4,4) = 7$ $J_B(4,4) = 5$

Again though this game matrix differs from that given in Table 2.2 by only a single objective value, $J_A(2,2) = 10 \Rightarrow 8$; this game matrix does not contain a single set of Nash strategies. Without such an equilibrium point, it is again difficult to determine a "good" strategy.

2.2 STACKELBERG EQUILIBRIUM STRATEGIES

Stackelberg type strategies are an alternative popular game theoretic equilibrium concept. Recall that Stackelberg's work examined the case in which one player, the leader, had the initiative while the other player, the follower, can only react. Here the leader has the ability to choose its strategy before the follower. Assuming both players are intelligent, only the leader is truly capable of deciding its destiny as the follower should naturally react optimally to the leaders

announced strategy. Consequently, in cases where the leader knows the objective function of the follower, the leader is capable of determining the corresponding optimal strategy of the follower for each of the follower's possible strategies. This implies that the leader can determine the objective function value resulting from for each of its possible strategies and the corresponding optimal response of the follower. The Stackelberg strategy can then be defined as the strategy which results in the best objective score for the leader, given an optimal reaction by the follower. A mathematical definition of the Stackelburg strategy was first given as follows [23]:

$$R_B\left(\hat{u}_A\right) = u_B^* \text{ such that } J_B\left(\hat{u}_A, u_B^*\right) \geq J_B\left(\hat{u}_A, u_B\right) \quad \forall u_B \in U_B \qquad \text{and} \tag{2.2a}$$

$$R_A\left(\hat{u}_B\right) = u_A^* \text{ such that } J_A\left(u_A^*, \hat{u}_B\right) \geq J_A\left(u_A, \hat{u}_B\right) \quad \forall u_A \in U_A \tag{2.2b}$$

then the Stackelberg strategies, u_A^S and u_B^S with Player A as leader and Player B as follower, can be defined as

$$J_A\left(u_A^S, R_B\left(u_A^S\right)\right) \geq J_A\left(u_A, R_B\left(u_A\right)\right) \quad \forall u_A \in U_A. \tag{2.2c}$$

$$J_B\left(R_A\left(u_B^S\right), u_B^S\right) \geq J_B\left(R_A\left(u_B\right), u_B\right) \quad \forall u_B \in U_B. \tag{2.2d}$$

In cases where the strategy spaces are finite, the Stackelberg strategy can also be found by means of a game matrix [23]. For example, consider the game matrix shown in Table 2.5 in which Player B is the leader. To find a Stackelberg strategy, Player B must first find the optimal response of Player A for each possible strategy of team B. For each of these strategies, Player B must then determine the resulting value of its own objective function. In the case of Table 2.5 these objective scores correspond to (15,2,3,3) if Player B's strategies are scanned left to right. Consequently, the Stackelberg strategy is $u_B^S = 1$, as this strategy results in the best outcome for Player B given that Player A selects its optimal response.

Table 2.5 – Finding a Stackelberg Strategy Through the Use of a Game Matrix

	$u_B = 1$	$u_B = 2$	$u_B = 3$	$u_B = 4$
$u_A = 1$	$J_A^*(1,1) = 5$ $J_B(1,1) = 15$	$J_A(1,2) = 6$ $J_B(1,2) = 4$	$J_A(1,3) = 14$ $J_B(1,3) = 1$	$J_A(1,4) = 0$ $J_B(1,4) = 20$
$u_A = 2$	$J_A(2,1) = 1$ $J_B(2,1) = 7$	$J_A(2,2) = 8$ $J_B(2,2) = 8$	$J_A(2,3) = 4$ $J_B(2,3) = 4$	$J_A(2,4) = 6$ $J_B(2,4) = 2$
$u_A = 3$	$J_A(3,1) = 4$ $J_B(3,1) = 6$	$J_A(3,2) = 2$ $J_B(3,2) = 7$	$J_A^*(3,3) = 20$ $J_B(3,3) = 3$	$J_A^*(3,4) = 8$ $J_B(3,4) = 3$
$u_A = 4$	$J_A(4,1) = 3$ $J_B(4,1) = 10$	$J_A^*(4,2) = 9$ $J_B(4,2) = 2$	$J_A(4,3) = 16$ $J_B(4,3) = 5$	$J_A(4,4) = 7$ $J_B(4,4) = 5$

2.3 APPLYING GAME THEORETIC CONCEPTS TO A MT-DWTA EXAMPLE

To illustrate the applicability of game theoretic strategy concepts to the MT-DWTA we will consider a simple example. In particular, we will apply the concept of Nash equilibrium to a MT-WTA example of the type given in (1.6). Consider a 2 battle-step problem involving a Red and a Blue team where both the Red and Blue teams are each composed of two non-homogeneous units, i.e. $N_R = N_B = 2$. Assume that the probabilities of kill relating each of the four units are as follows:

$$\begin{array}{cc} p_{1,1}^R = .90 & p_{1,2}^R = .80 \\ p_{2,1}^R = .25 & p_{2,2}^R = .15 \\ p_{1,1}^B = .40 & p_{1,2}^B = .75 \\ p_{2,1}^B = .50 & p_{2,2}^B = .80 \end{array} \quad \therefore \quad \begin{array}{cc} q_{1,1}^R = .10 & q_{1,2}^R = .20 \\ q_{2,1}^R = .75 & q_{2,2}^R = .85 \\ q_{1,1}^B = .60 & q_{1,2}^B = .25 \\ q_{2,1}^B = .50 & q_{2,2}^B = .20 \end{array}. \tag{2.3}$$

Furthermore, assume that each of the two units on each team may only launch a single weapon per battle-step so that the constraints given in (1.7) become:

$$\begin{aligned} &\sum_{k=1}^{2}\sum_{j=1}^{2} u_{i,j}^{B}(k) \le 2, \ \ \forall i = 1,2 \\ &\sum_{k=1}^{2}\sum_{i=1}^{2} u_{j,i}^{R}(k) \le 2, \ \ \forall j = 1,2 \end{aligned}, \tag{2.4a}$$

$$\begin{aligned} &\sum_{i=1}^{2} u_{j,i}^{R}(k) \le 1, \ \ \forall i = 1,2 \\ &\sum_{j=1}^{2} u_{i,j}^{B}(k) \le 1, \ \ \forall j = 1,2 \end{aligned} \quad \text{and} \tag{2.4b}$$

$$\begin{aligned} &u_{i,j}^{B}(k) \ge 0 \text{ and is an integer } \forall i = 1,2, \ \ j = 1,2 \text{ and } k = 1,2 \\ &u_{j,i}^{B}(k) \ge 0 \text{ and is an integer } \forall i = 1,2, \ \ j = 1,2 \text{ and } k = 1,2 \end{aligned}. \tag{2.4c}$$

Because each unit may only target a single enemy unit per battle step, a target assignment strategy may be written more simply as $v_i^B(k)$ and $v_j^R(k)$, or the target of the i^{th} and j^{th} blue and red units respectively k^{th} battle step. Thus the target assignment strategy of each team at each time k can be written as:

$$\begin{aligned} v^B(k) &= \left\{ v_1^B(k), v_2^B(k), \ldots, v_{N_B}^B(k) \right\} \\ v^R(k) &= \left\{ v_1^R(k), v_2^R(k), \ldots, v_{N_R}^R(k) \right\} \end{aligned} \tag{2.5a}$$

and likewise the target assignment strategy of each team can be written over all time K as:

$$\begin{aligned} v^B &= \left\{ v^B(1); v^B(2); \ldots; v^B(K) \right\} \\ v^R &= \left\{ v^R(1); v^R(2); \ldots; v^R(K) \right\} \end{aligned}. \tag{2.5b}$$

For the purpose of constructing a game matrix, first consider an expected outcome matrix composed of row indexes representing each blue target assignment strategy v_B, eg {1,2;2,1}, and

each column index representing a red target assignment strategy v_R such that the expected outcome matrix contains all possible combinations of target assignment strategies.

		Red Weapon Target Assignment Strategies																
		{1,1;1,1}	{1,2;1,1}	{2,1;1,1}	{2,2;1,1}	{1,1;1,2}	{1,2;1,2}	{2,1;1,2}	{2,2;1,2}	{1,1;2,1}	{1,2;2,1}	{2,1;2,1}	{2,2;2,1}	{1,1;2,2}	{1,2;2,2}	{2,1;2,2}	{2,2;2,2}	
Blue Weapon Target Assignment Strategies	{1,1;1,1}	0.15 1.00 0.04 1.00	0.17 1.00 0.05 0.85	0.19 1.00 0.41 0.20	0.16 1.00 0.55 0.17	0.15 1.00 0.05 0.85	0.17 1.00 0.07 0.72	0.19 1.00 0.55 0.17	0.16 1.00 0.73 0.14	0.15 1.00 0.06 0.76	0.17 1.00 0.08 0.65	0.19 1.00 0.56 0.15	0.16 1.00 0.75 0.13	0.15 1.00 0.08 0.65	0.17 1.00 0.10 0.55	0.19 1.00 0.75 0.13	0.16 1.00 1.00 0.11	Red 1 Red 2 Blue 1 Blue 2
	{1,2;1,1}	0.29 0.20 0.03 1.00	0.33 0.20 0.04 0.85	0.38 0.20 0.33 0.20	0.33 0.20 0.44 0.17	0.29 0.20 0.03 0.97	0.33 0.20 0.05 0.82	0.38 0.20 0.35 0.19	0.33 0.20 0.46 0.16	0.29 0.20 0.07 0.52	0.33 0.20 0.10 0.44	0.38 0.20 0.71 0.10	0.33 0.20 0.95 0.09	0.29 0.20 0.08 0.50	0.33 0.20 0.10 0.43	0.38 0.20 0.75 0.10	0.33 0.20 1.00 0.09	Red 1 Red 2 Blue 1 Blue 2
	{2,1;1,1}	0.24 0.25 0.04 1.00	0.28 0.25 0.05 0.85	0.32 0.25 0.39 0.20	0.27 0.25 0.52 0.17	0.24 0.25 0.04 0.96	0.28 0.25 0.06 0.82	0.32 0.25 0.41 0.19	0.27 0.25 0.55 0.16	0.24 0.25 0.07 0.60	0.28 0.25 0.09 0.51	0.32 0.25 0.70 0.12	0.27 0.25 0.94 0.10	0.24 0.25 0.08 0.58	0.28 0.25 0.10 0.49	0.32 0.25 0.75 0.12	0.27 0.25 1.00 0.10	Red 1 Red 2 Blue 1 Blue 2
	{2,2;1,1}	0.49 0.05 0.01 1.00	0.55 0.05 0.01 0.85	0.63 0.05 0.07 0.20	0.55 0.05 0.10 0.17	0.49 0.05 0.01 0.99	0.55 0.05 0.01 0.84	0.63 0.05 0.08 0.20	0.55 0.05 0.10 0.17	0.49 0.05 0.07 0.20	0.55 0.05 0.10 0.17	0.63 0.05 0.74 0.04	0.55 0.05 0.99 0.03	0.49 0.05 0.08 0.20	0.55 0.05 0.10 0.17	0.63 0.05 0.75 0.04	0.55 0.05 1.00 0.03	Red 1 Red 2 Blue 1 Blue 2
	{1,1;1,2}	0.29 0.20 0.04 1.00	0.29 0.32 0.05 0.85	0.21 0.84 0.41 0.20	0.18 0.86 0.55 0.17	0.29 0.20 0.05 0.85	0.29 0.32 0.07 0.72	0.21 0.84 0.55 0.17	0.18 0.86 0.73 0.14	0.29 0.20 0.06 0.76	0.29 0.32 0.08 0.65	0.21 0.84 0.56 0.15	0.18 0.86 0.75 0.13	0.29 0.20 0.08 0.65	0.29 0.32 0.10 0.55	0.21 0.84 0.75 0.13	0.18 0.86 1.00 0.11	Red 1 Red 2 Blue 1 Blue 2
	{1,2;1,2}	0.58 0.04 0.03 1.00	0.58 0.06 0.04 0.85	0.42 0.17 0.33 0.20	0.36 0.17 0.44 0.17	0.58 0.04 0.03 0.97	0.58 0.06 0.05 0.82	0.42 0.17 0.35 0.19	0.36 0.17 0.46 0.16	0.58 0.04 0.07 0.52	0.58 0.06 0.10 0.44	0.42 0.17 0.71 0.10	0.36 0.17 0.95 0.09	0.58 0.04 0.08 0.50	0.58 0.06 0.10 0.43	0.42 0.17 0.75 0.10	0.36 0.17 1.00 0.09	Red 1 Red 2 Blue 1 Blue 2
	{2,1;1,2}	0.49 0.05 0.04 1.00	0.48 0.08 0.05 0.85	0.35 0.21 0.39 0.20	0.30 0.22 0.52 0.17	0.49 0.05 0.04 0.96	0.48 0.08 0.06 0.82	0.35 0.21 0.41 0.19	0.30 0.22 0.55 0.16	0.49 0.05 0.07 0.60	0.48 0.08 0.09 0.51	0.35 0.21 0.70 0.12	0.30 0.22 0.94 0.10	0.49 0.05 0.08 0.58	0.48 0.08 0.10 0.49	0.35 0.21 0.75 0.12	0.30 0.22 1.00 0.10	Red 1 Red 2 Blue 1 Blue 2
	{2,2;1,2}	0.97 0.01 0.01 1.00	0.96 0.02 0.01 0.85	0.70 0.04 0.07 0.20	0.60 0.04 0.10 0.17	0.97 0.01 0.01 0.99	0.96 0.02 0.01 0.84	0.70 0.04 0.08 0.20	0.60 0.04 0.10 0.17	0.97 0.01 0.07 0.20	0.96 0.02 0.10 0.17	0.70 0.04 0.74 0.04	0.60 0.04 0.99 0.03	0.97 0.01 0.08 0.20	0.96 0.02 0.10 0.17	0.70 0.04 0.75 0.04	0.60 0.04 1.00 0.03	Red 1 Red 2 Blue 1 Blue 2
	{1,1;2,1}	0.15 0.94 0.04 1.00	0.17 0.93 0.05 0.85	0.27 0.44 0.41 0.20	0.27 0.25 0.55 0.17	0.15 0.94 0.05 0.85	0.17 0.93 0.07 0.72	0.27 0.44 0.55 0.17	0.27 0.25 0.73 0.14	0.15 0.94 0.06 0.76	0.17 0.93 0.08 0.65	0.27 0.44 0.56 0.15	0.27 0.25 0.75 0.13	0.15 0.94 0.08 0.65	0.17 0.93 0.10 0.55	0.27 0.44 0.75 0.13	0.27 0.25 1.00 0.11	Red 1 Red 2 Blue 1 Blue 2
	{1,2;2,1}	0.30 0.19 0.03 1.00	0.35 0.19 0.04 0.85	0.54 0.09 0.33 0.20	0.55 0.05 0.44 0.17	0.30 0.19 0.03 0.97	0.35 0.19 0.05 0.82	0.54 0.09 0.35 0.19	0.55 0.05 0.46 0.16	0.30 0.19 0.07 0.52	0.35 0.19 0.10 0.44	0.54 0.09 0.71 0.10	0.55 0.05 0.95 0.09	0.30 0.19 0.08 0.50	0.35 0.19 0.10 0.43	0.54 0.09 0.75 0.10	0.55 0.05 1.00 0.09	Red 1 Red 2 Blue 1 Blue 2
	{2,1;2,1}	0.25 0.24 0.04 1.00	0.29 0.23 0.05 0.85	0.45 0.11 0.39 0.20	0.46 0.06 0.52 0.17	0.25 0.24 0.04 0.96	0.29 0.23 0.06 0.82	0.45 0.11 0.41 0.19	0.46 0.06 0.55 0.16	0.25 0.24 0.07 0.60	0.29 0.23 0.09 0.51	0.45 0.11 0.70 0.12	0.46 0.06 0.94 0.10	0.25 0.24 0.08 0.58	0.29 0.23 0.10 0.49	0.45 0.11 0.75 0.12	0.46 0.06 1.00 0.10	Red 1 Red 2 Blue 1 Blue 2
	{2,2;2,1}	0.50 0.05 0.01 1.00	0.58 0.05 0.01 0.85	0.90 0.02 0.07 0.20	0.92 0.01 0.10 0.17	0.50 0.05 0.01 0.99	0.58 0.05 0.01 0.84	0.90 0.02 0.08 0.20	0.92 0.01 0.10 0.17	0.50 0.05 0.07 0.20	0.58 0.05 0.10 0.17	0.90 0.02 0.74 0.04	0.92 0.01 0.99 0.03	0.50 0.05 0.08 0.20	0.58 0.05 0.10 0.17	0.90 0.02 0.75 0.04	0.92 0.01 1.00 0.03	Red 1 Red 2 Blue 1 Blue 2
	{1,1;2,2}	0.30 0.19 0.04 1.00	0.30 0.30 0.05 0.85	0.30 0.37 0.41 0.20	0.30 0.22 0.55 0.17	0.30 0.19 0.05 0.85	0.30 0.30 0.07 0.72	0.30 0.37 0.55 0.17	0.30 0.22 0.73 0.14	0.30 0.19 0.06 0.76	0.30 0.30 0.08 0.65	0.30 0.37 0.56 0.15	0.30 0.22 0.75 0.13	0.30 0.19 0.08 0.65	0.30 0.30 0.10 0.55	0.30 0.37 0.75 0.13	0.30 0.22 1.00 0.11	Red 1 Red 2 Blue 1 Blue 2
	{1,2;2,2}	0.60 0.04 0.03 1.00	0.60 0.06 0.04 0.85	0.60 0.07 0.33 0.20	0.60 0.04 0.44 0.17	0.60 0.04 0.03 0.97	0.60 0.06 0.05 0.82	0.60 0.07 0.35 0.19	0.60 0.04 0.46 0.16	0.60 0.04 0.07 0.52	0.60 0.06 0.10 0.44	0.60 0.07 0.71 0.10	0.60 0.04 0.95 0.09	0.60 0.04 0.08 0.50	0.60 0.06 0.10 0.43	0.60 0.07 0.75 0.10	0.60 0.04 1.00 0.09	Red 1 Red 2 Blue 1 Blue 2
	{2,1;2,2}	0.50 0.05 0.04 1.00	0.50 0.07 0.05 0.85	0.50 0.09 0.39 0.20	0.50 0.05 0.52 0.17	0.50 0.05 0.04 0.96	0.50 0.07 0.06 0.82	0.50 0.09 0.41 0.19	0.50 0.05 0.55 0.16	0.50 0.05 0.07 0.60	0.50 0.07 0.09 0.51	0.50 0.09 0.70 0.12	0.50 0.05 0.94 0.10	0.50 0.05 0.08 0.58	0.50 0.07 0.10 0.49	0.50 0.09 0.75 0.12	0.50 0.05 1.00 0.10	Red 1 Red 2 Blue 1 Blue 2
	{2,2;2,2}	1.00 0.01 0.01 1.00	1.00 0.01 0.01 0.85	1.00 0.02 0.07 0.20	1.00 0.01 0.10 0.17	1.00 0.01 0.01 0.99	1.00 0.01 0.01 0.84	1.00 0.02 0.08 0.20	1.00 0.01 0.10 0.17	1.00 0.01 0.07 0.20	1.00 0.01 0.10 0.17	1.00 0.02 0.74 0.04	1.00 0.01 0.99 0.03	1.00 0.01 0.08 0.20	1.00 0.01 0.10 0.17	1.00 0.02 0.75 0.04	1.00 0.01 1.00 0.03	Red 1 Red 2 Blue 1 Blue 2

Figure 2.1 – Example of an Expected Outcome Matrix for a 2x2 MT-DWTA

Each combination of target assignment strategies, in this case 16x16=256 possibilities, yields a probability of survival for each unit according to (1.6). Accordingly, each entry in the expected

outcome matrix will have four different probabilities of survival, R1, R2, B1 and B2 as shown in Figure 2.1 and listed $\left[R_1\, R_2\, B_1\, B_2\right]^{\mathrm{T}}$ in each cell.

Blue Weapon Target Assignment Strategies	Red Weapon Target Assignment Strategies															
	{1,1;1,1}	{1,2;1,1}	{2,1;1,1}	{2,2;1,1}	{1,1;1,2}	{1,2;1,2}	{2,1;1,2}	{2,2;1,2}	{1,1;2,1}	{1,2;2,1}	{2,1;2,1}	{2,2;2,1}	{1,1;2,2}	{1,2;2,2}	{2,1;2,2}	{2,2;2,2}
{1,1;1,1}	0.10	0.26	0.58	0.45	0.24	0.37	0.47	0.29	0.33	0.44	0.47	0.29	0.42	0.52	0.31	0.05
	-0.10	-0.26	-0.58	-0.45	-0.24	-0.37	-0.47	-0.29	-0.33	-0.44	-0.47	-0.29	-0.42	-0.52	-0.31	-0.05
{1,2;1,1}	-0.54	-0.36	0.05	-0.08	-0.51	-0.34	0.04	-0.10	-0.10	-0.01	-0.24	-0.51	-0.09	0.00	-0.27	-0.56
	0.54	0.36	-0.05	0.08	0.51	0.34	-0.04	0.10	0.10	0.01	0.24	0.51	0.09	0.00	0.27	0.56
{2,1;1,1}	-0.55	-0.38	-0.02	-0.16	-0.51	-0.35	-0.04	-0.19	-0.18	-0.08	-0.26	-0.52	-0.16	-0.06	-0.30	-0.57
	0.55	0.38	0.02	0.16	0.51	0.35	0.04	0.19	0.18	0.08	0.26	0.52	0.16	0.06	0.30	0.57
{2,2;1,1}	-0.47	-0.26	0.41	0.33	-0.47	-0.25	0.41	0.33	0.26	0.33	-0.10	-0.42	0.26	0.33	-0.11	-0.43
	0.47	0.26	-0.41	-0.33	0.47	0.25	-0.41	-0.33	-0.26	-0.33	0.10	0.42	-0.26	-0.33	0.11	0.43
{1,1;1,2}	-0.55	-0.30	0.44	0.33	-0.41	-0.19	0.33	0.17	-0.33	-0.11	0.34	0.16	-0.23	-0.04	0.17	-0.07
	0.55	0.30	-0.44	-0.33	0.41	0.19	-0.33	-0.17	0.33	0.11	-0.34	-0.16	0.23	0.04	-0.17	0.07
{1,2;1,2}	-0.41	-0.25	0.06	-0.07	-0.38	-0.23	0.05	-0.09	0.03	0.10	-0.23	-0.51	0.04	0.11	-0.26	-0.55
	0.41	0.25	-0.06	0.07	0.38	0.23	-0.05	0.09	-0.03	-0.10	0.23	0.51	-0.04	-0.11	0.26	0.55
{2,1;1,2}	-0.50	-0.34	-0.03	-0.17	-0.47	-0.31	-0.05	-0.20	-0.14	-0.04	-0.26	-0.52	-0.12	-0.03	-0.31	-0.58
	0.50	0.34	0.03	0.17	0.47	0.31	0.05	0.20	0.14	0.04	0.26	0.52	0.12	0.03	0.31	0.58
{2,2;1,2}	-0.03	0.12	0.47	0.37	-0.02	0.12	0.47	0.37	0.71	0.71	-0.04	-0.38	0.71	0.71	-0.05	-0.39
	0.03	-0.12	-0.47	-0.37	0.02	-0.12	-0.47	-0.37	-0.71	-0.71	0.04	0.38	-0.71	-0.71	0.05	0.39
{1,1;2,1}	0.05	0.19	0.10	-0.19	0.19	0.30	-0.01	-0.35	0.28	0.38	-0.01	-0.35	0.37	0.45	-0.17	-0.59
	-0.05	-0.19	-0.10	0.19	-0.19	-0.30	0.01	0.35	-0.28	-0.38	0.01	0.35	-0.37	-0.45	0.17	0.59
{1,2;2,1}	-0.54	-0.36	0.10	-0.01	-0.52	-0.34	0.09	-0.03	-0.10	-0.01	-0.19	-0.44	-0.09	0.00	-0.22	-0.49
	0.54	0.36	-0.10	0.01	0.52	0.34	-0.09	0.03	0.10	0.01	0.19	0.44	0.09	0.00	0.22	0.49
{2,1;2,1}	-0.55	-0.38	-0.03	-0.17	-0.52	-0.35	-0.05	-0.19	-0.18	-0.09	-0.26	-0.52	-0.17	-0.07	-0.31	-0.58
	0.55	0.38	0.03	0.17	0.52	0.35	0.05	0.19	0.18	0.09	0.26	0.52	0.17	0.07	0.31	0.58
{2,2;2,1}	-0.46	-0.24	0.65	0.66	-0.45	-0.23	0.65	0.66	0.27	0.35	0.14	-0.09	0.27	0.35	0.13	-0.11
	0.46	0.24	-0.65	-0.66	0.45	0.23	-0.65	-0.66	-0.27	-0.35	-0.14	0.09	-0.27	-0.35	-0.13	0.11
{1,1;2,2}	-0.55	-0.31	0.06	-0.20	-0.42	-0.20	-0.05	-0.36	-0.33	-0.13	-0.05	-0.36	-0.23	-0.05	-0.21	-0.59
	0.55	0.31	-0.06	0.20	0.42	0.20	0.05	0.36	0.33	0.13	0.05	0.36	0.23	0.05	0.21	0.59
{1,2;2,2}	-0.40	-0.23	0.15	0.04	-0.37	-0.21	0.13	0.02	0.05	0.12	-0.14	-0.40	0.06	0.13	-0.18	-0.44
	0.40	0.23	-0.15	-0.04	0.37	0.21	-0.13	-0.02	-0.05	-0.12	0.14	0.40	-0.06	-0.13	0.18	0.44
{2,1;2,2}	-0.49	-0.33	0.01	-0.13	-0.46	-0.30	-0.01	-0.16	-0.12	-0.03	-0.23	-0.49	-0.11	-0.02	-0.27	-0.54
	0.49	0.33	-0.01	0.13	0.46	0.30	0.01	0.16	0.12	0.03	0.23	0.49	0.11	0.02	0.27	0.54
{2,2;2,2}	0.00	0.15	0.74	0.74	0.01	0.16	0.74	0.74	0.74	0.75	0.24	-0.01	0.74	0.75	0.23	-0.02
	0.00	-0.15	-0.74	-0.74	-0.01	-0.16	-0.74	-0.74	-0.74	-0.75	-0.24	0.01	-0.74	-0.75	-0.23	0.02

Figure 2.2 – Example of a Game Matrix for the 2x2 MT-DWTA

Next, evaluating the red and blue objective functions given in (1.8) yields the game matrix for this problem. For simplicity, we will assume that each Red and Blue unit are of unit worth to each team, thus $c_{B1}^{B} = c_{B2}^{B} = c_{R1}^{B} = c_{R2}^{B} = 1$ and $c^{B} = c^{R}$. Figure 2.2 depicts the resulting game

matrix, with each entry in the form $\begin{matrix} J_R\left(v_B, v_R\right) \\ J_B\left(v_B, v_R\right) \end{matrix}$. Following the construction of the game matrix it is possible to solve for the Nash strategies. As in the example given in Table 2.2, the optimal target assignment strategy for Blue, highlighted in blue, is determined for each possible target assignment strategy of the Red team. Similarly, this process is repeated to yield the optimal target assignment strategy for Red, highlighted in red, for each possible Blue target assignment strategy. According to (2.1), matrix entries in the game matrix shown in Figure 2.2 that exist in both Red's and Blue's reaction sets, i.e. entries containing both red and blue highlights, are therefore Nash solutions. Using this criterion to investigate the game matrix in shown in Figure 2.2 discloses two Nash solutions,

$$\begin{aligned} v_R^{N_1} &= [2,1;1,1], v_B^{N_1} = [2,1;1,2] \\ v_R^{N_2} &= [2,1;1,1], v_B^{N_2} = [2,1;2,1] \end{aligned} \tag{2.6a}$$

where

$$\begin{aligned} J_R\left(v_R^{N_1}, v_B^{N_1}\right) &= -.03, J_B\left(v_R^{N_1}, v_B^{N_1}\right) = .03 \\ J_R\left(v_R^{N_2}, v_B^{N_2}\right) &= -.03, J_B\left(v_R^{N_2}, v_B^{N_2}\right) = .03 \end{aligned}. \tag{2.6b}$$

As previously discussed, a game matrix having two Nash solutions generates problems for game theoretic approaches as it becomes possible for each team to choose a different Nash solution. Nevertheless, because there is only one distinct Nash target assignment strategy for the red team and the objective function outcomes are the same for each of the two blue Nash strategies, this problem is inconsequential. This can be seen in the game matrix shown in Figure 2.3.

		Red Weapon Target Assignment Strategies															
		{1,1;1,1}	{1,2;1,1}	{2,1;1,1}	{2,2;1,1}	{1,1;1,2}	{1,2;1,2}	{2,1;1,2}	{2,2;1,2}	{1,1;2,1}	{1,2;2,1}	{2,1;2,1}	{2,2;2,1}	{1,1;2,2}	{1,2;2,2}	{2,1;2,2}	{2,2;2,2}
Blue Weapon Target Assignment Strategies	{1,1;1,1}	0.10 -0.10	0.26 -0.26	0.58 -0.58	0.45 -0.45	0.24 -0.24	0.37 -0.37	0.47 -0.47	0.29 -0.29	0.33 -0.33	0.44 -0.44	0.47 -0.47	0.29 -0.29	0.42 -0.42	0.52 -0.52	0.31 -0.31	0.05 -0.05
	{1,2;1,1}	-0.54 0.54	-0.36 0.36	0.05 -0.05	-0.08 0.08	-0.51 0.51	-0.34 0.34	0.04 -0.04	-0.10 0.10	-0.10 0.10	-0.01 0.01	-0.24 0.24	-0.51 0.51	-0.09 0.09	0.00 0.00	-0.27 0.27	-0.56 0.56
	{2,1;1,1}	-0.55 0.55	-0.38 0.38	-0.02 0.02	-0.16 0.16	-0.51 0.51	-0.35 0.35	-0.04 0.04	-0.19 0.19	-0.18 0.18	-0.08 0.08	-0.26 0.26	-0.52 0.52	-0.16 0.16	-0.06 0.06	-0.30 0.30	-0.57 0.57
	{2,2;1,1}	-0.47 0.47	-0.26 0.26	0.41 -0.41	0.33 -0.33	-0.47 0.47	-0.25 0.25	0.41 -0.41	0.33 -0.33	0.26 -0.26	0.33 -0.33	-0.10 0.10	-0.42 0.42	0.26 -0.26	0.33 -0.33	-0.11 0.11	-0.43 0.43
	{1,1;1,2}	-0.55 0.55	-0.30 0.30	0.44 -0.44	0.33 -0.33	-0.41 0.41	-0.19 0.19	0.33 -0.33	0.17 -0.17	-0.33 0.33	-0.11 0.11	0.34 -0.34	0.16 -0.16	-0.23 0.23	-0.04 0.04	0.17 -0.17	-0.07 0.07
	{1,2;1,2}	-0.41 0.41	-0.25 0.25	0.06 -0.06	-0.07 0.07	-0.38 0.38	-0.23 0.23	0.05 -0.05	-0.09 0.09	0.03 -0.03	0.10 -0.10	-0.23 0.23	-0.51 0.51	0.04 -0.04	0.11 -0.11	-0.26 0.26	-0.55 0.55
	{2,1;1,2}	-0.50 0.50	-0.34 0.34	-0.03 0.03	-0.17 0.17	-0.47 0.47	-0.31 0.31	-0.05 0.05	-0.20 0.20	-0.14 0.14	-0.04 0.04	-0.26 0.26	-0.52 0.52	-0.12 0.12	-0.03 0.03	-0.31 0.31	-0.58 0.58
	{2,2;1,2}	-0.03 0.03	0.12 -0.12	0.47 -0.47	0.37 -0.37	-0.02 0.02	0.12 -0.12	0.47 -0.47	0.37 -0.37	0.71 -0.71	0.71 -0.71	-0.04 0.04	-0.38 0.38	0.71 -0.71	0.71 -0.71	-0.05 0.05	-0.39 0.39
	{1,1;2,1}	0.05 -0.05	0.19 -0.19	0.10 -0.10	-0.19 0.19	0.19 -0.19	0.30 -0.30	-0.01 0.01	-0.35 0.35	0.28 -0.28	0.38 -0.38	-0.01 0.01	-0.35 0.35	0.37 -0.37	0.45 -0.45	-0.17 0.17	-0.59 0.59
	{1,2;2,1}	-0.54 0.54	-0.36 0.36	0.10 -0.10	-0.01 0.01	-0.52 0.52	-0.34 0.34	0.09 -0.09	-0.03 0.03	-0.10 0.10	-0.01 0.01	-0.19 0.19	-0.44 0.44	-0.09 0.09	0.00 0.00	-0.22 0.22	-0.49 0.49
	{2,1;2,1}	-0.55 0.55	-0.38 0.38	-0.03 0.03	-0.17 0.17	-0.52 0.52	-0.35 0.35	-0.05 0.05	-0.19 0.19	-0.18 0.18	-0.09 0.09	-0.26 0.26	-0.52 0.52	-0.17 0.17	-0.07 0.07	-0.31 0.31	-0.58 0.58
	{2,2;2,1}	-0.46 0.46	-0.24 0.24	0.65 -0.65	0.66 -0.66	-0.45 0.45	-0.23 0.23	0.65 -0.65	0.66 -0.66	0.27 -0.27	0.35 -0.35	0.14 -0.14	-0.09 0.09	0.27 -0.27	0.35 -0.35	0.13 -0.13	-0.11 0.11
	{1,1;2,2}	-0.55 0.55	-0.31 0.31	0.06 -0.06	-0.20 0.20	-0.42 0.42	-0.20 0.20	-0.05 0.05	-0.36 0.36	-0.33 0.33	-0.13 0.13	-0.05 0.05	-0.36 0.36	-0.23 0.23	-0.05 0.05	-0.21 0.21	-0.59 0.59
	{1,2;2,2}	-0.40 0.40	-0.23 0.23	0.15 -0.15	0.04 -0.04	-0.37 0.37	-0.21 0.21	0.13 -0.13	0.02 -0.02	0.05 -0.05	0.12 -0.12	-0.14 0.14	-0.40 0.40	0.06 -0.06	0.13 -0.13	-0.18 0.18	-0.44 0.44
	{2,1;2,2}	-0.49 0.49	-0.33 0.33	0.01 -0.01	-0.13 0.13	-0.46 0.46	-0.30 0.30	-0.01 0.01	-0.16 0.16	-0.12 0.12	-0.03 0.03	-0.23 0.23	-0.49 0.49	-0.11 0.11	-0.02 0.02	-0.27 0.27	-0.54 0.54
	{2,2;2,2}	0.00 0.00	0.15 -0.15	0.74 -0.74	0.74 -0.74	0.01 -0.01	0.16 -0.16	0.74 -0.74	0.74 -0.74	0.74 -0.74	0.75 -0.75	0.24 -0.24	-0.01 0.01	0.74 -0.74	0.75 -0.75	0.23 -0.23	-0.02 0.02

Figure 2.3 – Illustration of 2 Nash Strategy Pairs for a 2x2 MT-DWTA

2.4 SCALABILITY ISSUES INHERENT IN GAME THEORETIC APPROACHES

The simple example shown in Section 3 demonstrates the effectiveness of game theoretic approaches to the multi-team dynamic weapon target assignment problem. However, it also

illustrates that a game matrix approach to solving such problems can easily become unfeasible for even small numbers of units. The previous example considered only two Red and two Blue units over two battle steps. Nevertheless, this example generated 256 possible target assignment strategy combinations, 512 objective function scores and 1024 unit outcomes. If a more permissive case is considered, in which a unit is not forced to select a target, these numbers become much higher. Consider the general case of the MT-DWTA problem in which N_B units on team Blue are engaged with N_R units on team Red over K battle-steps. It is assumed that each of the N_B units on team Blue may target any of the N_R unit on team Red or abstain from targeting altogether. Hence each Blue unit may select from $\left(N_R+1\right)$ possible target assignment strategies at each battle step. This yields a total of $\left(N_R+1\right)^{N_B}$ target assignment strategies for the Blue team to consider at each battle step, or $\left(N_R+1\right)^{N_B K}$ target assignment strategies over the course of the entire battle. Applying this result to the red team yields a total of $\left(N_B+1\right)^{N_R K}$ possible strategies. This implies that the entire search space consists of a total of

$$\left(\left(N_B+1\right)^{N_R}\left(N_R+1\right)^{N_B}\right)^K \tag{2.7}$$

possible target assignment strategy combinations. If (2.7) is examined for several values of N_R, N_B and K, as shown in Table 2.6, it becomes apparent that it is not feasible to use a game matrix to solve for the Nash equilibrium strategies for even simple cases of the MT-DWTA. This scalability issue is the main deterrent to such approaches.

Table 2.6 – Size of the Search Space for Various MT-DTWA Problems

N_B	N_R	S_{Battle}	Size of Search Space	N_B	N_R	S_{Battle}	Size of Search Space
2	2	1	8.1000E+01	2	2	2	6.5610E+03
2	4	1	2.0250E+03	2	4	2	4.1006E+06
4	4	1	3.9063E+05	4	4	2	1.5259E+11
4	8	1	2.5629E+09	4	8	2	6.5684E+18
8	8	1	1.8530E+15	8	8	2	3.4337E+30
8	16	1	1.2926E+25	8	16	2	1.6709E+50
16	16	1	2.3679E+39	16	16	2	5.6070E+78
45	45	1	4.4484E+149	45	45	2	1.9788E+299

Though a game theoretic approach is a natural formulation from which to solve the multi-team weapon target assignment problem, a more efficient Nash solution search procedure is required before such an approach can be applied to MT-WTA problems of even moderate size. Consequently, in this dissertation the author will present and evaluate such a method. This method is based largely on a WTA assignment algorithm entitled the Unit Level Team Resource Allocation Algorithm, or ULTRA, which will also be defined and evaluated. The author will demonstrate that not only is the implementation of a game theoretic approach practical enough to be calculated real time, any team using such an approach will achieve a higher objective function score than a team considering a non-game, or "naive", approach regardless of the method that team's opponent uses to calculate its target assignment strategy.

3.0 FINDING AN OPTIMAL REACTION IN A MT-DWTA

The MT-DWTA is an important problem when considering many military conflicts. However, it is a difficult problem to implement in a practical manner. As mentioned previously, the competitive nature of the MT-DWTA prohibits the use of typical optimization techniques for even simple cases of the MT-DWTA. This is because the objective functions of each team are linked. Consequently, certain game theoretic concepts need to be integrated into standard optimization techniques. However, we previously demonstrated that it is computationally infeasible to solve the MT-DWTA using typical game theoretic methodology. The MT-DWTA yields a far too large of a game matrix when to be computationally feasible when considering even a small number of units. As a result, before the MT-DWTA problem can be practically applied to real world situations, an algorithm is needed that is capable of finding a game theoretic solution many orders of magnitude faster than by using a game matrix.

One possible approach is to separate the problem into a game theoretic problem and a standard optimization problem. To accomplish this, we will find the optimal target assignment strategy for a team when that team knows the target assignment strategies of its adversaries. While, this assumption is inherently untrue, it can be useful to solving the general problem. In chapter 4 we will show that it is possible to generate approximate target assignment strategies for MT-DWTA if a method is developed to quickly solve the general MT-DWTA when the target assignment strategies of all opposing teams are known.

In this chapter we first present a special case of the MT-DWTA, denoted the Single Team Response to the MT-DWTA (SMT-DWTA), in which a single team desires to find an optimal strategy knowing the target assignment strategies of the opposing teams. This formulation is identical to that of the general MT-DWTA except that the target assignment strategies, u^X, are known for all teams X except for the team in question, which we will refer to as team A. Then, we then describe a "good" target assignment strategy and derive performance measures to evaluate potential target assignment algorithms. After this, we will present an algorithm, denoted the Unit Level Team Resource Allocation algorithm (ULTRA), capable of quickly generating an approximate solution to the ST-DWTA. Finally, the performance of this algorithm will be evaluated using the previously defined performance measures.

3.1 GENERAL FORMULATION OF THE SMT-DWTA

In SMT-DWTA, we seek the optimal targets selection strategy for a team given the target assignment strategies of its adversaries, a certain attrition model and a particular objective function formulation. The SMT-DWTA naturally decomposes into these three basic concepts, target assignment strategies, attrition models and objective functions. These concepts are defined in this order. This is because an objective function can not be formulated without knowing the attrition model and an attrition model can not be expressed without first expressing the form of the target assignment strategies.

To formulate an expression for the target assignment strategies consider SMT-DWTA in its most general form. In the most general formulation, the SMT-DWTA consists of two or more

independent teams, each composed of a number of heterogeneous units, engaged in a multi-stage conflict broken into discrete battle steps. The units are each assumed to carry various numbers of non-homogeneous weapons that may be assigned to any unit on an opposing team according to that team's target assignment strategy. Consequently, a target assignment strategy is required to provide the number and types of weapons that each unit will expend as well as each of these weapon's corresponding target at each battle-step. These target assignment strategies are subject to a set of constraints representing the target assignment limitations inherent in the given conflict. For example, these constraints may limit the number of weapons that a unit may expend and/or the number of distinct targets a unit may assign weapons to at any given battle step. Certain units may also be unable to target certain opposing units due to range or sensor restrictions.

Expressed mathematically, consider again the general case in which each team X, where $X \in \{A, B, C, ...\}$, present in a conflict lasting K battle-steps is composed of N_X units. Each unit x on team X is armed with W_x^X types of weapons. Furthermore, a unit x is assumed to have a quantity of the ω^{th} type of weapon equal to $W_{x,\omega}^X$. The number of the ω^{th} type of weapon on unit x assigned to any unit on any opposing team at a battle-step k is denoted through the use of a target assignment strategy, $u_{x,\omega}^X(k)$. This target assignment strategy $u_{x,\omega}^X(k)$ is a vector composed of an entry for each opposing unit present in the conflict where $u_{x,\omega,y}^X(k) \geq 0$ for all units x, weapons ω, targets y and battle-steps k.

To further illustrate this concept, consider the case where the 2nd unit on team A possesses a quantity of the 3rd type of weapon at the 2nd battle-step. If there are two other teams present, B

and C, having two and three units respectively, then a sample target assignment strategy for a single weapon might be

$$u^A_{2,3}(2) = \left[u^A_{2,3,B_1}(2), u^A_{2,3,B_2}(2); u^A_{2,3,C_1}(2), u^A_{2,3,C_2}(2), u^A_{2,3,C_3}(2) \right]$$
$$u^A_{2,3}(2) = [0,3;1,0,1] \tag{3.1}$$

In this target assignment strategy, the second unit of team A will launch 3 weapons of type 3 at the third unit of team B and 1 weapon of type 3 at the first and third units of team C. The target assignment strategy for a single weapon may be combined to form the target assignment strategy of a single unit:

$$u^X_x(k) = \left[u^X_{x,1}(k), u^X_{x,2}(k), \ldots, u^X_{x,W^X_x}(k) \right] \tag{3.2a}$$

or the target assignment strategy of an entire team:

$$u^X(k) = \left[u^X_1(k), u^X_2(k), \ldots, u^X_{N_X}(k) \right] \tag{3.2b}$$

or even the target assignment strategy an entire team over all K battle-steps:

$$u^X(k) = \left[u^X(1), u^X(2), \ldots, u^X(K) \right]. \tag{3.2c}$$

These target assignment strategies are generally subject to a set of constraints. For example the number of weapons of type ω allocated by the x^{th} unit is limited to the total number equipped, $W^X_{x,w}$. Expressed mathematically:

$$\sum_{k=1}^{K} \sum_{y}^{\text{Opposing Units}} u^X_{x,\omega,y}(k) \le W^X_{x,\omega}. \tag{3.3a}$$

Additionally the number of weapons of a given type allocated by a single unit x could be limited for a single time step such that

$$\sum_{y}^{\text{Opposing Units}} u^X_{x,\omega,y}(k) \le w^X_{x,\omega} \quad \forall k. \tag{3.3b}$$

Another possible constraint is that a unit x may target a limited number of distinct targets at each battle-step:

$$\sum_{y}^{\text{Opposing Units}} \operatorname{sgn}\left(u^{X}_{x,\omega,y}(k)\right) \le t^{X}_{x} \quad \forall k \tag{3.3c}$$

Having developed a basic formulation for possible target assignment strategies it is now possible to consider a generalized formulation of the SMT-DWTA attrition model. The attrition model is necessarily a state based process that relates the status of the units present in the conflict from one battle step to the next given a set of target assignment strategies. The exact formulation of an attrition model is determined by the nature of the conflict but, in general the degradation of a given unit is a function of its capabilities, the capabilities of the attacking units, and the target assignment strategies of all teams present in the conflict.

We will define this degradation according to the "health" of a unit at the end of a battle step. There are two possible interpretations of the term health. On one hand, a unit's health may represent the probability that a unit it fully functional at the end of a battle-step. It may also represent the remaining percentage of that unit's initial capabilities. Mathematically, we can represent the health of the i^{th} unit on team A after the k^{th} battle-step as $A_i(k)$. Consequently, an attrition function can be generally defined as follows:

$$A_i(k) = \alpha^{A}\left(A(k-1), u^{A}(k), B(k-1), u^{B}(k), C(k-1), u^{B}(k), \ldots\right). \tag{3.4}$$

where $\alpha^{A}(\bullet)$ is a function of the target assignment strategies of all teams as well as the current measure of health for all units.

Each team present in the conflict must have an objective function based upon the attrition model which that team attempts to maximize by selecting a target assignment strategy. Any team engaging in such a conflict would necessarily desire to preserve its units while eliminating

any opposition. A team may also seek to preserve its own weapons and or to force an opponent to waste their arsenal. As such an objective function can be broken into a summation of objective functions, each corresponding to the status of the units of a given team. For example if a team X has weapons remaining at the end of a battle given by $W^X(k)$, which is a function of that team's target assignment strategies u^X, then the portion of the objective function of team A dependent on team X, denoted $J_{A,X}$, can be written as a function of the remaining weapons and units of team, $J_{A,X}\left(X(k),W^X(k)\right)$. These partial objectives may be combined to form an overall objective function formulation as follows:

$$J_A = \sum_{X=A}^{B,C,D,\ldots} J_{A,X}\left(X(k),W^X(k)\right). \tag{3.5a}$$

This formulation can be also written in terms of each teams target assignment strategies,

$$J_A = \sum_{X=A}^{B,C,D,\ldots} J_{A,X}\left(A(k-1),u^A(k),B(k-1),u^B(k),C(k-1),u^B(k),\ldots\right). \tag{3.5b}$$

Consequently, a general formulation of the SMT-DWTA in the case of team A optimally responding to a set of target assignment strategies from each of its adversaries can be expressed as

$$\max_{u^A} \sum_{X=A}^{B,C,D,\ldots} J_{A,X}\left(A(k-1),u^A(k),B(k-1),u^B(k),C(k-1),u^B(k),\ldots\right) \tag{3.6a}$$

subject to the set of constraints given in (3.3)

3.2 MEASURING THE PERFORMANCE OF SOLUTION METHODS

The SMT-DWTA is not a game theoretic problem. Consequently, it is capable of being solved using optimization techniques. Additionally, the SMT-DWTA is an approximate model of the realities of a conflict. These two properties make the SMT-DWTA solvable in two ways. On one hand, we can employ an optimization algorithm which exactly solves a SMT-DWTA instance. The only method known to solve the SMT-DWTA exactly is an exhaustive search. We could also utilize a much faster heuristic algorithm to generate an approximate solution. The large contingent of standard optimization methods coupled with the permissibility of non-optimal solutions allows an infinite number of solution methods. Consequently, we must establish valid solution guidelines in order to intelligently select an appropriate approach. Here we will make use of measures that compare the two most important measurable qualities, the speed and the accuracy of a given algorithm.

In any real life application of the MT-DWTA, a target assignment strategy must be found in a certain finite and limited amount of time. The optimal strategy is useless if it can not be found within the specified amount of time. This implies that processor time is the most important performance measure of any MT-DWTA algorithm. Consequently, a suboptimal target assignment strategy available when needed is far superior to an optimal target assignment strategy available only after the time which it is needed. When this property is coupled with the size, non-linearity and the number of constraints inherent in the STM-DWTA we can conclude that a heuristic approach may be preferable to an optimization based approach.

Another measure of great importance is the correctness of a given target assignment strategy. This is a measure of how close a given strategy is to the optimal strategy obtained by exhaustive search. The accuracy of an algorithm, $A_{Algorithm}$, can be defined as the average of the resultant objective function value of its target assignment strategy, $J_{Algorithm}$, divided by that of the exhaustive search, $J_{Optimal}$ as follows:

$$A_{Algorithm} = \left\langle \frac{J_{Algorithm}}{J_{Optimal}} \right\rangle * 100, \tag{3.7}$$

where $\langle \cdot \rangle$ is the mean operator and represents the average value over all possible scenarios. Unfortunately, this measure implies that the optimal target assignment strategy be known a priori. As proven earlier, the computational complexity of the MT-DWTA problem prohibits an exhaustive search for a moderate number of units. Consequently, the accuracy of an algorithm can be found exactly only in instances of the SMT-DWTA involving small numbers of units. We need to extrapolate this data from problems involving a small number of units to generate accuracy values from more complicated problems.

A last consideration of an algorithms performance that must be taken into consideration is consistency. It is not always possible to determine the best algorithm for a given SMT-DWTA based solely upon the expected accuracy and number of objective function evaluations of each algorithm. In some formulations, an algorithm might be preferred over another that is on average more accurate but less consistent. Therefore, other important performance measures include the minimum, maximum and the variance of an algorithm's accuracy as well as the corresponding number of objective function evaluations. Another measure of consistency to consider is the threshold performance of an algorithm. The threshold performance is can be defined as the probability that an algorithm achieves a given minimum accuracy. In this

dissertation, we consider thresholds of 100, 99, 95 and 90 percent accuracy. These combined measures provide a concise measure of the consistency of a given algorithm.

3.3 UNIT LEVEL TEAM RESOURCE ALLOCATION ALGORITHM (ULTRA)

One class of algorithms that have been successfully applied to complex non-linear problems such as the SMT-DWTA is a class of heuristic algorithms labeled as Large Scale Neighborhood Search algorithms (LSNS) [26]. LSNS algorithms define a class of algorithms that function by first selecting an initial feasible strategy. Algorithms of this class then find the optimal strategy of a small subsection selection of feasible strategies related to the initial strategy. These strategies are considered to be the "neighborhood" of the initial strategy. Once a LSNS has found the optimal strategy in the given neighborhood, a new neighborhood is created around that optimal neighbor. A LSNS algorithm then proceeds to iteratively find the optimal neighbors in a neighborhood until an optimal neighbor is the optimal neighbor in its own neighborhood.

To solve the SMT-DWTA we introduce a LSNS algorithm which we call the Unit Level Team Resource Allocation algorithm (ULTRA). As with all LSNS algorithms, the ULTRA algorithm can be decomposed into three separate modules. First, ULTRA will begin by generating an initial feasible target assignment strategy. Second, a variable sized neighborhood "surrounding" the initial target assignment strategy will be found according to a rule module. Third, ULTRA will proceed to locate the locally optimal strategy in the given neighborhood. ULTRA will then proceed, iteratively generating neighborhoods and locating locally optimal strategies until a strategy is found which is optimal in its given neighborhood. Figure 3.1

provides a basic flowchart of the ULTRA algorithm. It should be noted that it is not necessary to generate the entire neighborhood surrounding a given target assignment strategy providing that it is possible to consecutively scan through the entire neighborhood. In this manner, the next neighbor can be generated from the previous neighbor.

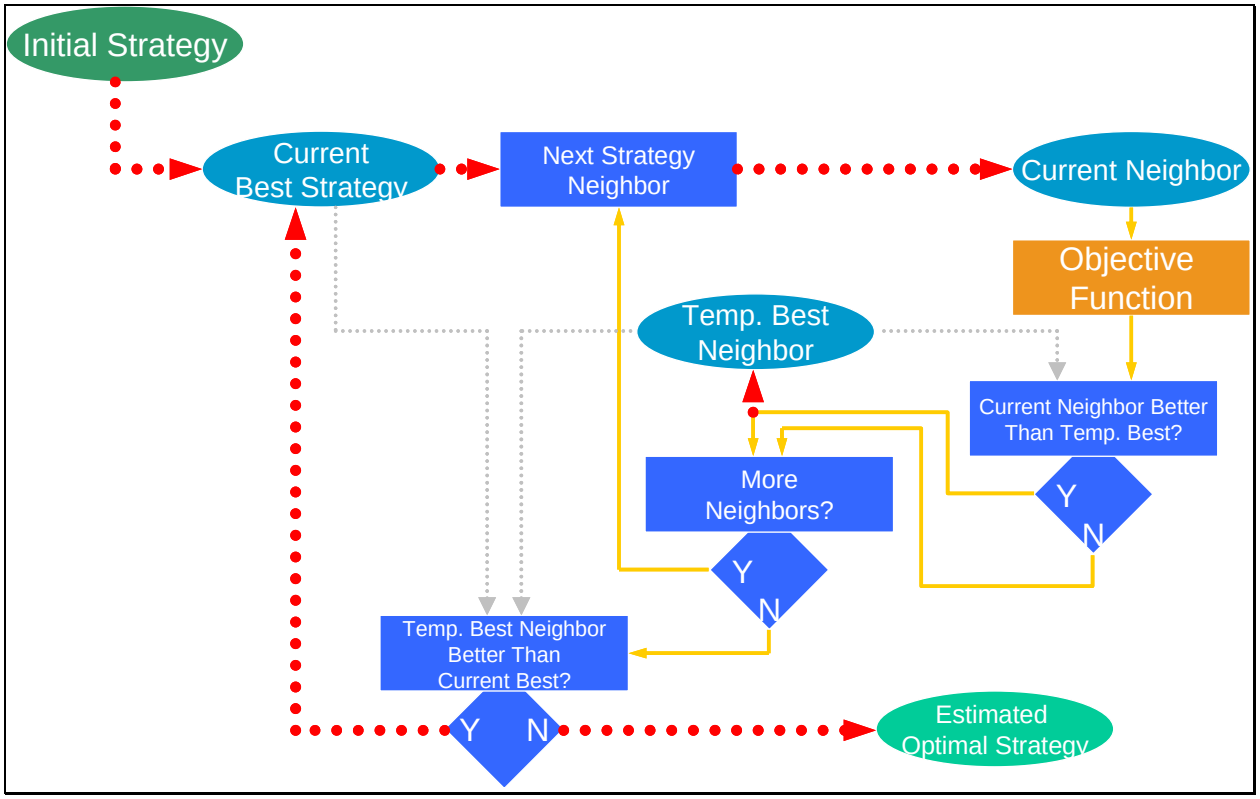

Figure 3.1 – ULTRA Functional Overview

3.3.1 ULTRA Neighborhood Search (F Degrees of Freedom)

At the heart of any LSNS is the neighborhood search. ULTRA uses a neighborhood search in which neighbors are defined according to the number of target assignment strategies differences. A difference occurs between two target assignment strategies when the same unit is allocated

different targets for at least one of its weapons for a given battle step. Target assignment strategies having F or fewer differences are assumed to be neighbors. We refer to this neighborhood search as the F degrees of freedom search. A flowchart is shown in Figure 3.2.

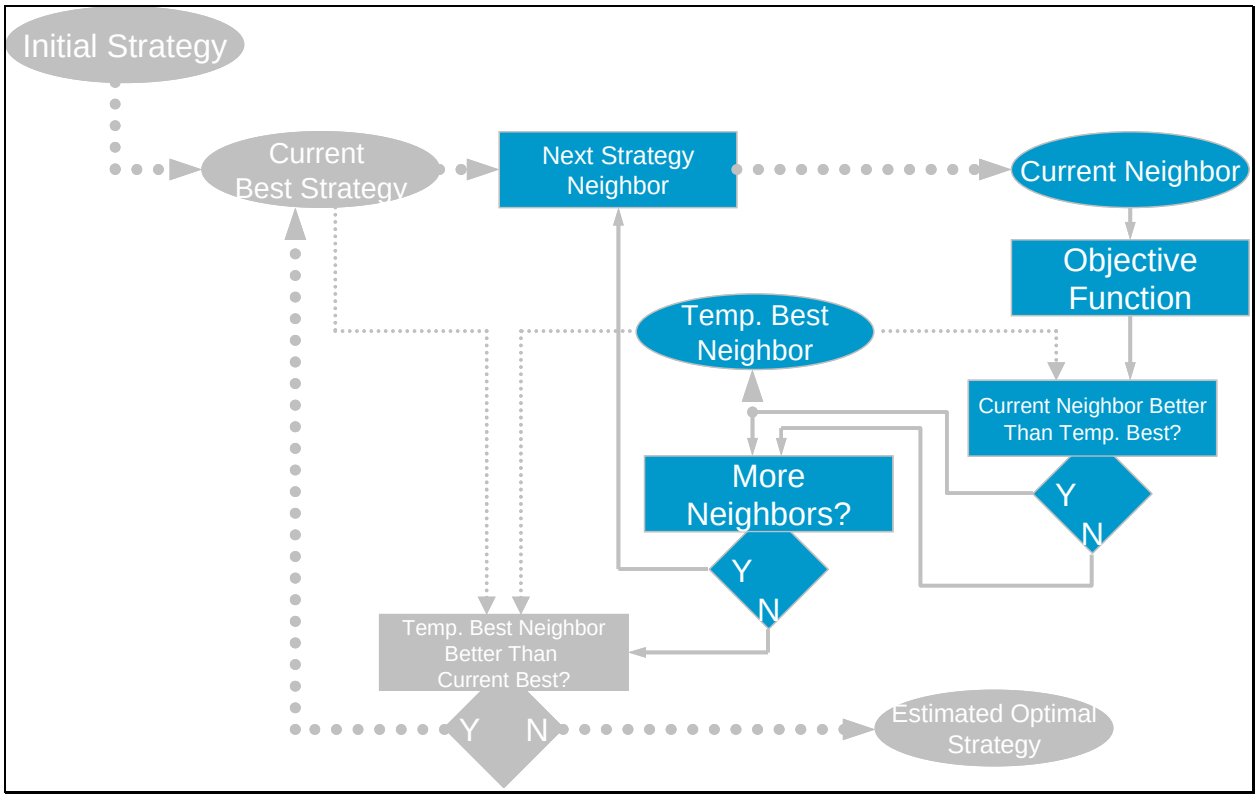

Figure 3.2 – Neighborhood Search in ULTRA

The F degrees of freedom search exhibits two qualities which qualify it as a well rounded neighborhood search. First, the F degrees of freedom neighborhood search is of variable complexity, enabling it to be "tuned" to provide optimal performance according to the given application. Second, the neighborhood can be constructed in such a manner as every neighbor can be generated sequentially, using only the previous neighbor. This means that ULTRA may

be implemented efficiently, using only a small amount of memory. This is an important consideration in cases when dealing with scenarios that generate very large neighborhoods.

As LSNS algorithms are typically non-deterministic, there is seldom any manner of determining whether or not the final local optimum strategy is equivalent to the global optimal strategy. One way to increase the chance that the final strategy will coincide with the optimum or to ensure a more optimal final strategy is to increase the size of the neighborhood. Conversely, these increases in the size of the neighborhood negatively impact the speed of the LSNS algorithm. If time is not of extreme importance, the degree of freedom coefficient, F, may be set high to achieve maximum accuracy. However, F may be set low for fastest results if the run time requirement of ULTRA is critical.

Another important consideration when implementing a LSNS algorithm is memory usage. Storing an entire neighborhood in memory is often impractical as a neighborhood can easily run into the hundreds of thousands of individual strategies. To resolve this problem, each neighbor can be generated from a previous neighbor in such a manner as to evaluate every neighbor once and only once. ULTRA accomplishes this through the use of counting modules. If each unit is labeled with a number from one to N, where N is the number of units, then it is possible to create a vector of F values representing the units which may vary their strategy from the given initial strategy. Such an implementation is illustrated in Figure 3.3.

To illustrate this search procedure further, consider an implementation of ULTRA on 5 units with a degree of freedom coefficient of 3. It is then possible to generate every possible combination of 3 of the 5 units allowed to change their strategies by "counting" in the following manner.

$$\begin{matrix}\{1,2,3\},\{1,2,4\},\{1,2,5\},\{1,3,4\},\{1,3,5\} \\ \{1,4,5\},\{2,3,4\},\{2,3,5\},\{2,4,5\},\{3,4,5\}\end{matrix} \tag{3.8}$$

Such a counting method is achieved by first creating a vector of size F to represent the units that may have their strategies changed as per the definition of an ULTRA neighbor. This vector is initially filled with units one through F. The counting algorithm then proceeds to increment by one the rightmost index having a value less than the total number of units minus F plus that index. Any indexes to the right of the index being updated are then set to one more than the index immediately left of itself. Similar counting type algorithms can also be used to search through all of the combinations of weapons and targets capable of being considered by each unit allowed to change its strategy in the given iteration. It can then be argued that by using such methods, all neighbors in a neighborhood can then be created and evaluated while only storing the initial strategy, the current best strategy in the given neighborhood and the current neighbor. This algorithm can be seen in Figure 3.3.

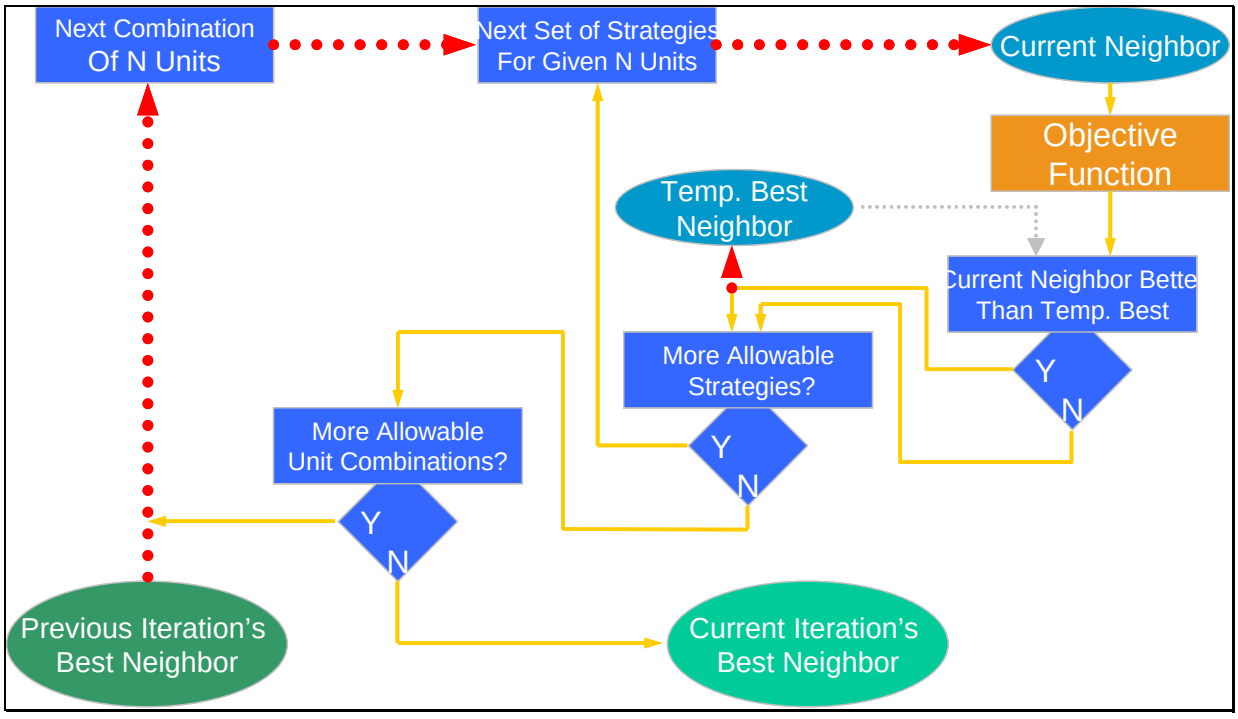

Figure 3.3 – F Degrees of Freedom Neighborhood Generation/Search

3.3.2 Iterative Best Neighbor Search

The outer loop of ULTRA is defined as an iterative best neighbor search. In this loop strategies are continuously improved by running a neighborhood search around a given strategy to find the neighborhood optimal strategy and then running a neighborhood optimality search around that strategy. This procedure continues until a neighborhood is generated around a strategy in which that strategy is the neighborhood optimal strategy.

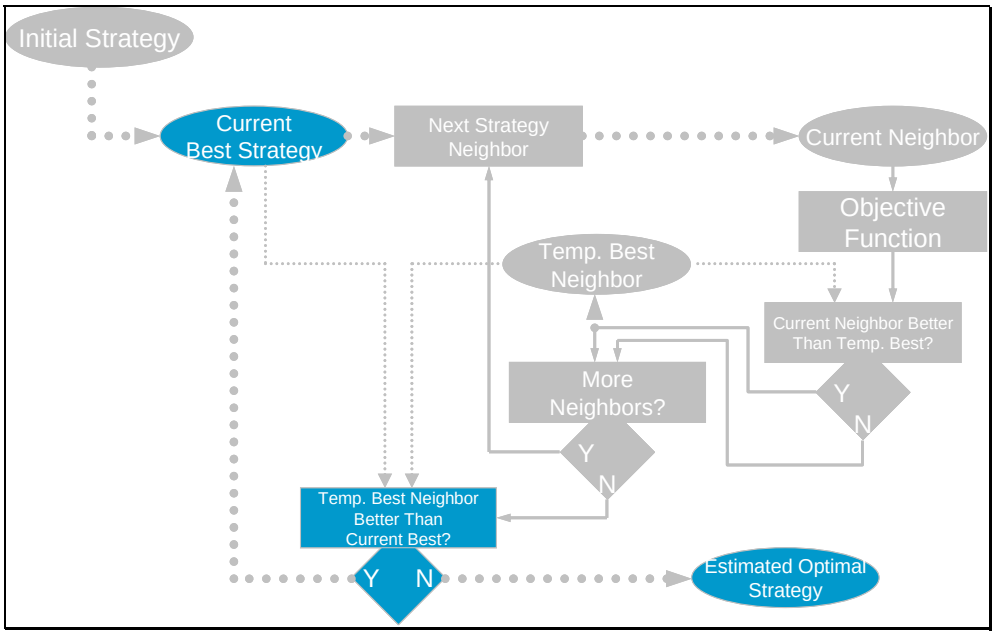

Figure 3.4 – Iterative Best Neighbor Search in ULTRA

Three features of the iterative best neighbor search should be noted. First, as the stopping criterion of the iterative best neighbor search requires that a strategy be optimal only in a neighborhood surrounding that strategy, this algorithm cannot be guaranteed to arrive at the globally optimal strategy in the general case. Second, because the objective function score of the

strategies considered in this algorithm are ever increasing it can be argued that ULTRA is likely to arrive at a "good" approximation of the optimum. Third, ULTRA is guaranteed to converge because the objective function value of the neighborhood strategy must increase from one iteration to the next and because there are assumed to be a finite number of possible strategies.

3.3.3 ULTRA Initial Condition

As with all LSNS algorithms the closer the initial strategy is to the globally optimal strategy, the better the chance that the LSNS algorithm will arrive at the global optimum. A well chosen initial strategy further aids the performance of the algorithm as it can enable the algorithm to converge faster. In this dissertation we will consider three different initial strategies, the zero strategy, the random strategy, and the unit greedy strategy.

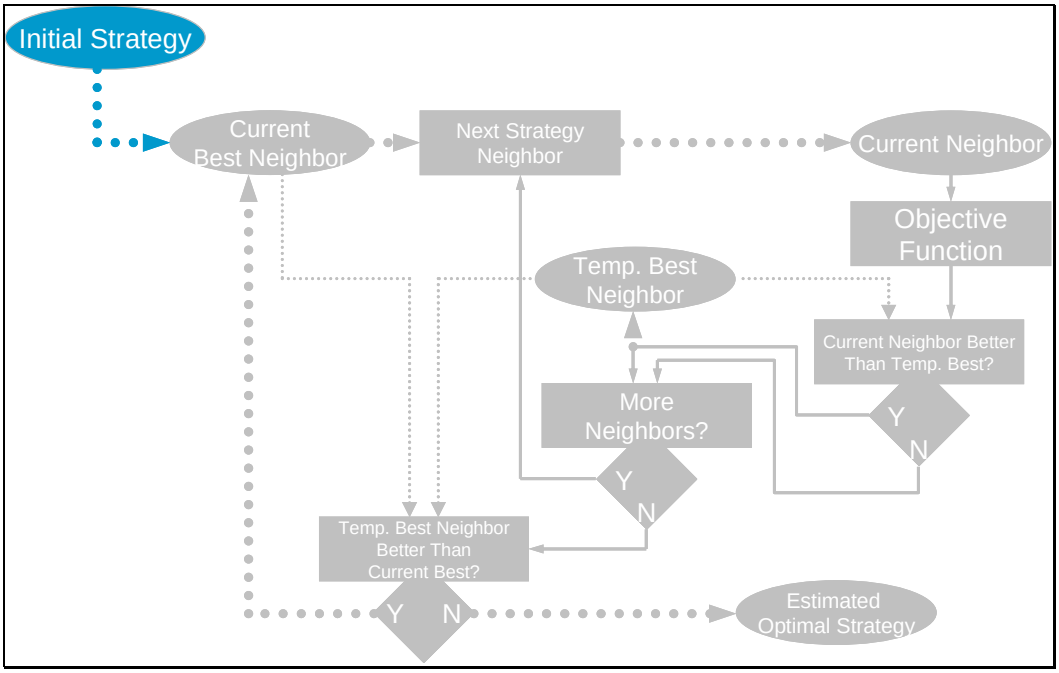

Figure 3.5 – The Initial Strategy in ULTRA

a) Definition 1: A vector $u_A^z \in U_A$ is called a **zero target assignment strategy** for Team A if each entry $u_{Ai}^z(k)$ of u_A^z is selected from the set $\{0\}$. Similarly, A vector $u_B^z \in U_B$ is called a **zero target assignment strategy** for Team B if each entry $u_{Bi}^z(k)$ of u_B^z is selected from the set $\{0\}$.

b) Definition 2: A vector $u_A^r \in U_A$ is called a **unit random strategy** for the Team A team if each entry $u_{Ai}^r(k)$ of u_A^r is selected randomly from the set $\{0,1,....,N_B\}$ with a certain probability distribution. Similarly, a vector $u_B^r \in U_B$ is called a **unit random strategy** for the Team B team if each entry $u_{Bi}^r(k)$ of u_B^r is selected randomly from the set $\{0,1,....,N_A\}$ with a certain probability distribution.

c) Definition 3: A vector $u_A^g \in U_A$ is called a **unit greedy strategy** for the Team A if each entry $u_{Ai}^g(k)$ in u_A^g is selected such that $J_A(\hat{u}_{Ai}^g(k), \underset{\sim}{0}, k) \geq J_A(\hat{u}_{Ai}^g(k), \underset{\sim}{0}, k)$ for $i = 1,2...,N_A$ and for all $\hat{u}_{Ai}(k) \in U_A$, where $\hat{u}_{Ai} = [0,..,0,u_{Ai},0,..,0]'$. Similarly, a vector $u_B^g \in U_B$ is called a **unit greedy strategy** for Team B if each entry $u_{Bi}^g(k)$ in u_B^g is selected such that $J_B(\hat{u}_{Bi}^g(k), \underset{\sim}{0}, k) \geq J_B(\hat{u}_{Bi}^g(k), \underset{\sim}{0}, k)$ for $i = 1,2...,N_B$ and for all $\hat{u}_{Bi}(k) \in U_B$, where $\hat{u}_{Bi} = [0,..,0,u_{Bi},0,..,0]'$.

These definitions can be described as follows; the zero strategy assumes that each weapon on each team is initially not given a target. In the case of the random strategy, a random task selection strategy is selected from the list of all feasible task selection strategies as the initial strategy. Lastly, the unit greedy strategy sets an initial strategy in which each unit is assigned the

task selection strategy which maximizes that unit's contribution to the overall team objective function assuming all other units are set to the zero strategy.

3.3.4 Illustration of ULTRA with F=1 on a Sample Scenario

To illustrate the mechanism of the ULTRA algorithm, we will walk through a simple example of the SMT-DTWA using a simple implementation of ULTRA with the degree of freedom coefficient set as 1. Recall the sample game matrix for a combat scenario involving two teams of two units over two battle steps given in Figure 3.2. To comply with the assumptions of the SMT-DWTA we will assume that the Blue team has already selected a task selection strategy, say for example the Nash strategy given by $u^B = \{2,1;2,1\}$. Thus the game matrix can be reduced to the following Figure:

Red Weapon Target Assignment Strategies																
	{1,1;1,1}	{1,2;1,1}	{2,1;1,1}	{2,2;1,1}	{1,1;1,2}	{1,2;1,2}	{2,1;1,2}	{2,2;1,2}	{1,1;2,1}	{1,2;2,1}	{2,1;2,1}	{2,2;2,1}	{1,1;2,2}	{1,2;2,2}	{2,1;2,2}	{2,2;2,2}
{2,1;2,1}	-0.55 0.55	-0.38 0.38	-0.03 0.03	-0.17 0.17	-0.52 0.52	-0.35 0.35	-0.05 0.05	-0.19 0.19	-0.18 0.18	-0.09 0.09	-0.26 0.26	-0.52 0.52	-0.17 0.17	-0.07 0.07	-0.31 0.31	-0.58 0.58

Figure 3.6 – Simple SMT-DWTA Example

where the highlighted entry represents the previously determined optimal Red strategy. It should be noted that the zero strategy has been omitted from this example due to spatial concerns. Continuing with this example, the first procedure undertaken by ULTRA is to select an initial strategy. In this case we will set the initial strategy as $u^R\{1,1;1,1\}$. Using this initial strategy,

ULTRA must now generate a neighborhood of strategies with a degree of freedom coefficient of one. This operation generates the following five neighbors:

$$\{1,1;1,1\},\{2,1;1,1\},\{1,2;1,1\},\{1,1;2,1\},\{1,1;1,2\}, \tag{3.9}$$

shown on the game matrix given in Figure 3.7:

	Red Weapon Target Assignment Strategies															
	{1,1;1,1}	{1,2;1,1}	{2,1;1,1}	{2,2;1,1}	{1,1;1,2}	{1,2;1,2}	{2,1;1,2}	{2,2;1,2}	{1,1;2,1}	{1,2;2,1}	{2,1;2,1}	{2,2;2,1}	{1,1;2,2}	{1,2;2,2}	{2,1;2,2}	{2,2;2,2}
{2,1;2,1}	-0.55 0.55	-0.38 0.38	-0.03 0.03	-0.17 0.17	-0.52 0.52	-0.35 0.35	-0.05 0.05	-0.19 0.19	-0.18 0.18	-0.09 0.09	-0.26 0.26	-0.52 0.52	-0.17 0.17	-0.07 0.07	-0.31 0.31	-0.58 0.58

Figure 3.7 – ULTRA First Iteration

Of these five choices, the target assignment strategy $u^R = \{2,1;1,1\}$ results in the highest objective function value. This strategy also happens to be the optimal strategy. Because the initial strategy was not the optimal in its own neighborhood, ULTRA begins the second iteration by generating a neighborhood around the current best neighbor, $u^R = \{2,1;1,1\}$. This second neighborhood is composed of the following 5 strategies:

$$\{2,1;1,1\},\{1,1;1,1\},\{2,2;1,1\},\{2,1;2,1\},\{2,1;1,2\} \tag{3.10}$$

in matrix form as shown in Figure 3.8:

	Red Weapon Target Assignment Strategies															
	{1,1;1,1}	{1,2;1,1}	{2,1;1,1}	{2,2;1,1}	{1,1;1,2}	{1,2;1,2}	{2,1;1,2}	{2,2;1,2}	{1,1;2,1}	{1,2;2,1}	{2,1;2,1}	{2,2;2,1}	{1,1;2,2}	{1,2;2,2}	{2,1;2,2}	{2,2;2,2}
{2,1;2,1}	-0.55 0.55	-0.38 0.38	-0.03 0.03	-0.17 0.17	-0.52 0.52	-0.35 0.35	-0.05 0.05	-0.19 0.19	-0.18 0.18	-0.09 0.09	-0.26 0.26	-0.52 0.52	-0.17 0.17	-0.07 0.07	-0.31 0.31	-0.58 0.58

Figure 3.8 – ULTRA Second Iteration

In the second iteration, the best neighbor is the same as the best neighbor in the previous iteration. This is the stopping criteria for ULTRA. Consequently, the target assignment strategy $u^R = \{2,1;1,1\}$ is returned as ULTRA's approximation of the globally optimal strategic response.

To contrast this result, we will quickly step through the previous example with a different initial strategy. In this case we will assume an initial strategy $u^R = \{1,2;2,2\}$. Generating the first neighborhood:

$$\{1,2;2,2\},\{2,2;2,2\},\{1,1;2,2\},\{1,2;1,2\},\{1,2;2,1\}. \tag{3.11}$$

Examining the SMT-DWTA matrix given in Figure 3.6 we find that the optimal neighbor in the set shown in (3.11) is the initial strategy $u^R = \{1,2;2,2\}$. This implies that ULTRA has converged and will return $u^R = \{1,2;2,2\}$ as the ULTRA approximation. Although this is not the globally optimal strategy it is a close approximation. The two resulting objective function values are -.07 and -.03 respectively. This is a good approximation considering the other possible values.

3.3.5 ULTRA Computational Complexity

In many instances, a SMT-DWTA is formulated in such a way as to provide a set amount of time in which a target assignment strategy must be chosen. In such cases it is implied that any target assignment algorithm should find the most optimal strategy possible given the provided amount of time. Recall that when dealing with LSNS type optimization algorithms, a larger neighborhood typically implies a more optimal strategy. Because ULTRA is "tunable," the degree of freedom coefficient should be set to the highest value that permits ULTRA to converge

in a prescribed amount of time. To determine the optimal value of F given a maximum number of objective function evaluations we must first generate an expression that approximates the number of objective function evaluations to F and the parameters of the SMT-DWTA.

To determine the number of objective function evaluations as a function of F and SMT-DWTA parameters we will examine the computational requirements of ULTRA while considering a general example. Consider a team of N units, each having W weapons of which w may be launched in each of K battle-steps, engaged in a conflict with M adversarial targets. The first quantity that must be found is the number of possible target assignment strategies available to each unit. It is assumed that every weapon on each unit may target each of the M targets or abstain from combat. As such, each weapon scheduled to be launched during a given battle step has $(M+1)$ possible target assignment strategies. This implies that a set of w weapons generates $(M+1)^w$ strategies. Every unit is capable of launching w weapons at each battle step implying that each unit may select from $(M+1)^w$ target assignment strategies for every possible permutation of selecting w of W weapons. Thus the total number of target assignment strategies available to a given unit, TSS_U, can be found using the well known statistical formulation

$$\binom{W}{w} = \frac{W!}{(W-w)!w!}, \tag{3.12}$$

implying that

$$TSS_U = \frac{W!(M+1)^w}{(W-w)!w!}. \tag{3.13}$$

It should be noted that (3.13) simplifies to

$$TSS_U = (M+1)^w \tag{3.14}$$

when $W = w$ and further to $TSS_U = M+1$ when $W = w = 1$. To find the number of target section strategies in a given neighborhood, TSS_N, first recall that a neighborhood is defined as the set of target assignment strategies differing from an initial target assignment strategy in the target assignments for F or fewer units. If each of these units is allowed to change to any of TSS_U possible target assignment strategies, then F units combine for a total of

$$\left(\frac{W!(M+1)^w}{(W-w)!w!} \right)^F \tag{3.15}$$

possible target assignment strategies. The size of a neighborhood in a given best neighbor search can then be found by multiplying (3.15) by the total number of permutations possible selecting F target assignment strategies from the number of units multiplied by the number of battle-steps, NK, as follows:

$$TSS_N = \left(\frac{NK!}{(NK-F)!F!} \right) \left(\frac{W!(M+1)^w}{(W-w)!w!} \right)^F . \tag{3.16}$$

The final step to determining the number of objective function evaluations, OFE, required by ULTRA for a given set of parameters, is to multiply the size of the neighborhood evaluated during each neighborhood search by the total number of neighborhood searches. As is typically the case for neighborhood search algorithms, it is not possible to exactly determine the total number of neighborhood searches. This implies that it is not possible to determine the number of iterations required by ULTRA to find a target assignment strategy. Nevertheless, it is possible to determine the minimum number of iterations required assuming that each unit must be assigned a target assignment strategy at each battle-step and no unit is initially assigned a strategy. During

each best neighbor search only F of the NK total target assignment strategies may be assigned at each best neighbor search. This necessitates a minimum of NK/F iterations. Further assuming that the total number of iterations required for ULTRA to converge is on the order of this minimum, an expression can then be formed for the order of the number of objective function evaluations required as follows:

$$o\{OFE\} = o\left\{\left(\frac{NK}{F}\right)\left(\frac{NK!}{(NK-F)!F!}\right)\left(\frac{W!(M+1)^w}{(W-w)!w!}\right)^F\right\}. \tag{3.17}$$

Another quantity of interest is a factor representing the number of objective function evaluations ULTRA will save as compared to exhaustive search. The size of the exhaustive search can be found by setting $F = NK$, thereby increasing the size of the neighborhood to the entire strategy space. This assumption results in the following expression for the total size of the target assignment strategy space:

$$TSS_E = \left(\frac{W!(M+1)^w}{(W-w)!w!}\right)^{NK}. \tag{3.18}$$

Using the previously derived expression for the number of objective function evaluations required by ULTRA shown in (3.17) as well as the expression given in (3.18), a ratio can be formed relating the exhaustive search and ULTRA.

$$\frac{TSS_E}{o\{OFE\}} = \frac{\left(\frac{W!(M+1)^w}{(W-w)!w!}\right)^{NK}}{o\left\{\left(\frac{NK}{F}\right)\left(\frac{NK!}{(NK-F)!F!}\right)\left(\frac{W!(M+1)^w}{(W-w)!w!}\right)^F\right\}}. \tag{3.19a}$$

$$o\left\{\frac{TSS_E}{OFE}\right\} = o\left\{\frac{\left(\frac{W!(M+1)^w}{(W-w)!w!}\right)^{NK-F}}{\left(\frac{NK}{F}\right)\left(\frac{NK!}{(NK-F)!F!}\right)}\right\} \tag{3.19b}$$

If only one type of weapon is allotted to each unit and the number of units and targets are related such that $N = M + 1$, the ratio given in (3.19) can be reduced to the following:

$$o\left\{\frac{TSS_E}{OFE}\right\} = o\left\{\frac{FF!N^{NK-F-1}}{K\left(\frac{NK!}{(NK-F)!}\right)}\right\}, \tag{3.20}$$

This formulation is reasonable because substituting NK for F in (3.20) reduces the expression to one. We can arrive at the following expressions given in (3.21) by substituting (3.17) for various values of F:

$$\begin{aligned} o\left\{\frac{TSS_E}{OFE}(F=1)\right\} &= o\left\{\frac{(N)^{NK-3}}{K^2}\right\} \\ o\left\{\frac{TSS_E}{OFE}(F=2)\right\} &= o\left\{\frac{4(N)^{NK-5}}{K^2\left(K-\frac{1}{N}\right)}\right\} \\ o\left\{\frac{TSS_E}{OFE}(F=3)\right\} &= o\left\{\frac{18(N)^{NK-7}}{K^2\left(K-\frac{1}{N}\right)\left(K-\frac{2}{N}\right)}\right\} \end{aligned} \quad . \tag{3.21}$$

Because $\frac{1}{N} \ll K$ for large N, when N is large and F is small the relationship given in (3.21) can be approximated as:

$$o\left\{\frac{TSS_E}{OFE}(F)\right\} \approx o\left\{\frac{F(F!)(N)^{NK-2F-1}}{K^{F+1}}\right\}. \tag{3.22}$$

To further illustrate the reduction in the required number of objective function evaluations consider the following Figure which plots the objective function evaluations required to find a single reaction versus the number of units on the given team for a single battle step when each $M+1=N$ and each unit possesses exactly one weapon.

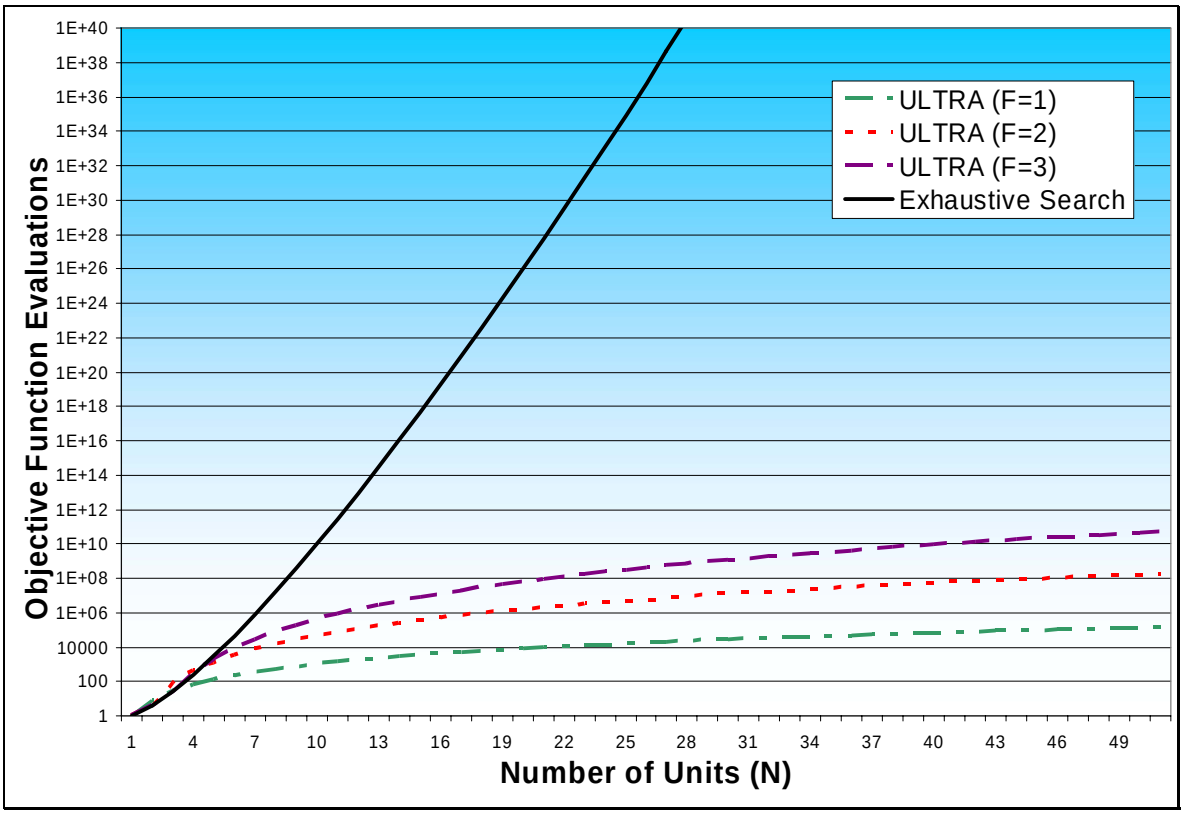

Figure 3.9 – Objective Function Evaluations Comparison

3.4 PERFORMANCE OF ULTRA ON A SAMPLE SMT-DWTA

In this section we will obtain valid performance measures for ULTRA. As we have previously mentioned, LSNS type algorithms are non-deterministic. While it often possible to mathematically bound the difference between the estimated and global optimum, this bound is not necessarily representative of actual algorithm performance. Consequently, the results of a LSNS algorithm cannot be known exactly without actually implementing the algorithm. Thus, ULTRA needs to be implemented on simulated scenarios to obtain valid performance measures. As simulations can only generate relevant performance measurements on problems similar to those simulated, a good method to gather valid performance measures is to average performance measures over many instances of a general simulation with random parameters. Thus, we will introduce a general scenario with random parameters. We will then simulate various instances of the general scenario, collecting aggregate measures.

To test the performance of the ULTRA algorithm we propose the following scenario with random parameters. Consider the case in which Team A, composed of N_A units is responding to a known strategy of Team B, composed of N_B units. For the sake of simplicity, we will assume that the objective function model is of the type given in (1.8) and evaluated over a single battle step using the six criteria. First, to generate aggregate performance measures, we will evaluate

this simulation 25000[1] times using both ULTRA and the exhaustive search. Second, the probabilities of kill were randomized each individual run. At any given run we independently generated each probability of kill using uniformly distributed random numbers over the interval $\left[0, P_i^A\right]$ and for Team A. It should be noted that it was not necessary to generate probability of kill values for Team B as the optimization is uncoupled for $K=1$[2]. Third, we assume that the worth of each unit to each team is one, or $c_{B1}^B = c_{B2}^B = c_{R1}^B = c_{R2}^B = 1$. Fourth, we conducted experiments for many different combinations of numbers of units on each team. Because we are comparing the ULTRA algorithm to the exhaustive search, we note that it not possible to consider more complicated situations than approximately six units versus six units. In such cases the exhaustive search becomes computationally unfeasible. Fifth, we evaluated the performance of the ULTRA algorithm for several different values of the degree of freedom coefficient, $F \in \{1,2,3\}$. Sixth, we considered three possible initial conditions for the ULTRA algorithm which we defined previously, the zero initial target assignment strategy, the random initial target assignment strategy and the unit greedy target assignment strategy.

Using these criteria we conducted three experiments to calculate the performance measures given in Section 3.2 and to determine how these performance measures vary as the various parameters of the problem change. In the first experiment, we examined how the performance of the algorithm varies with the number of units per side when each side has an equal number of units. In the second experiment we examined the effect that the initial target assignment strategy has on the overall performance of the ULTRA algorithm. In the third experiment we considered

[1] We chose the number 25,000 experimentally; the aggregate values began to form smooth curves after on the order of 15,000 runs. The extra 10,000 runs were completed to ensure a more accurate depiction of the algorithms performance.

[2] This relationship between coupled and uncoupled objective functions in the MT-DWTA is further explored in Chapter 5.

the effect of a dissimilar number of units on each team. This included varying the number of units on Team A while the number of units on Team B remain constant, varying the number of units on Team B while maintaining the same number of units on Team A and varying the number of units on both teams simultaneously. Again we emphasize that in all cases all measurements refer to Team A's optimization

It should be noted that the simplified SMT-DWTA model tested in the following experiments represents a valid approximation to the overall SMT-DWTA problem for two reasons. First, allowing units to fire a single weapon is a valid simplification because a unit with multiple weapons can be represented as multiple units with a single weapon when evaluated over a single battle step. Second, due to the inherent limitation of the exhaustive search, it is not possible to evaluate cases of the SMT-DWTA composed of more than seven or eight units for more than a single battle step.

3.4.1 Experiment I

In the first experiment, we modeled the effect of the number of units per side when all sides have an equal number of units. Here we assumed a zero target assignment strategy as the initial target assignment strategy. Using the performance measures given in Section 3.2 we evaluated the test scenario for the cases when $N_A = N_B = \{1,2,3,4,5,6\}$ for $F = \{1,2\}$. For each scenario considered, the model was simulated 25,000 times, with each instance having randomly generated probability of kill values such that $P_i^B = 1 \; \forall i \in \{1,2,\ldots,N_B\}$.

One important performance measure mentioned earlier is the average accuracy. This measurement plots the average percentage of the objective function value returned by the

ULTRA algorithm as compared to the optimal objective function value. Figure 3.10 plots the results of the average accuracy of the ULTRA algorithm versus the number of units per team. Figure 3.10 shows that when *F=1*, there is an exponentially asymptotic relationship between the number of units per side and the expected accuracy of the algorithm. The smoothness of the curve allows for a reasonable interpolation that the average expected accuracy for very large numbers of units is approximately 94%. Figure 3.10 also demonstrates that setting the degree of freedom coefficient to *F=2* results in a considerable improvement in the expected accuracy of the algorithm.

Another important characterization of any heuristic algorithm how well it performs in a worst case scenario. Figure 3.11 plots the worst accuracy measure encountered in the 25,000 instances evaluated for each of the number of units and degrees of freedom listed previously.

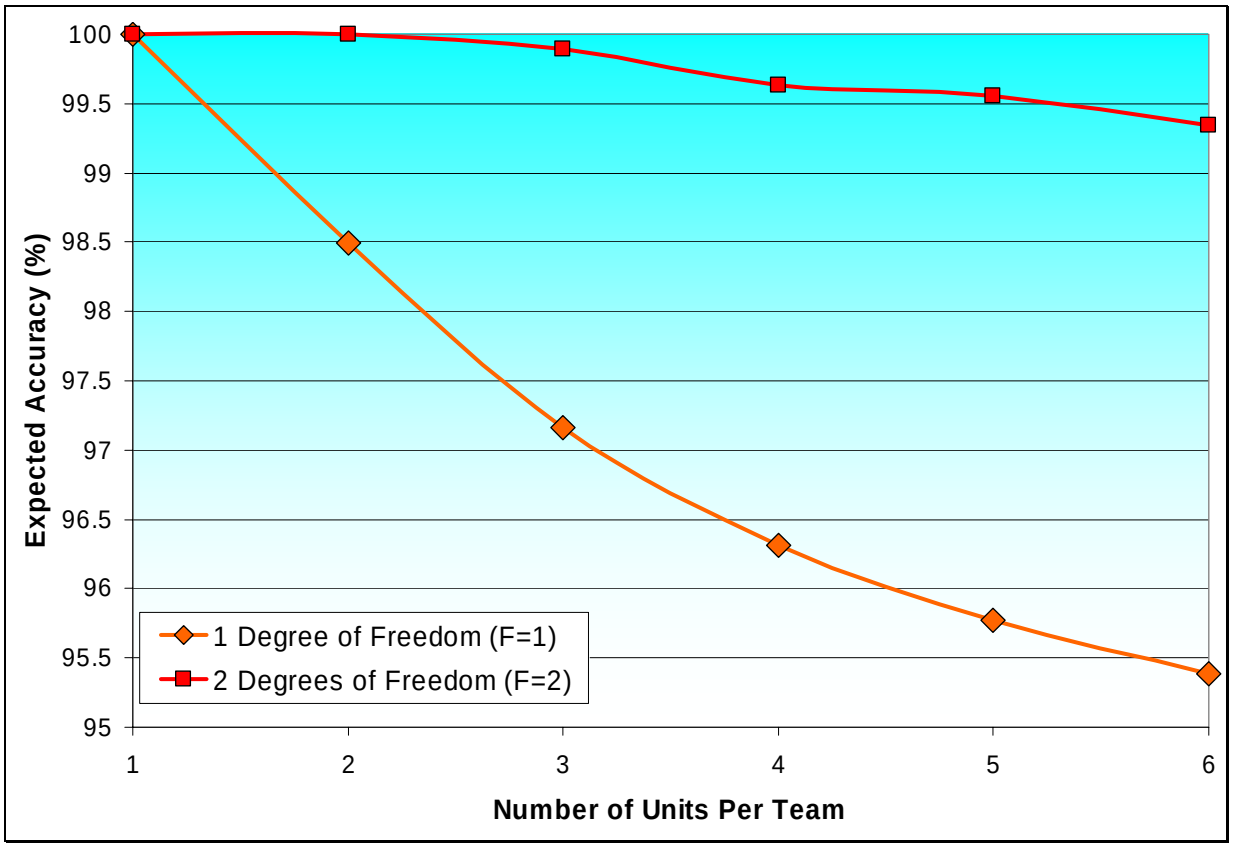

Figure 3.10 – Average Accuracy of ULTRA vs. Number of Units per Team

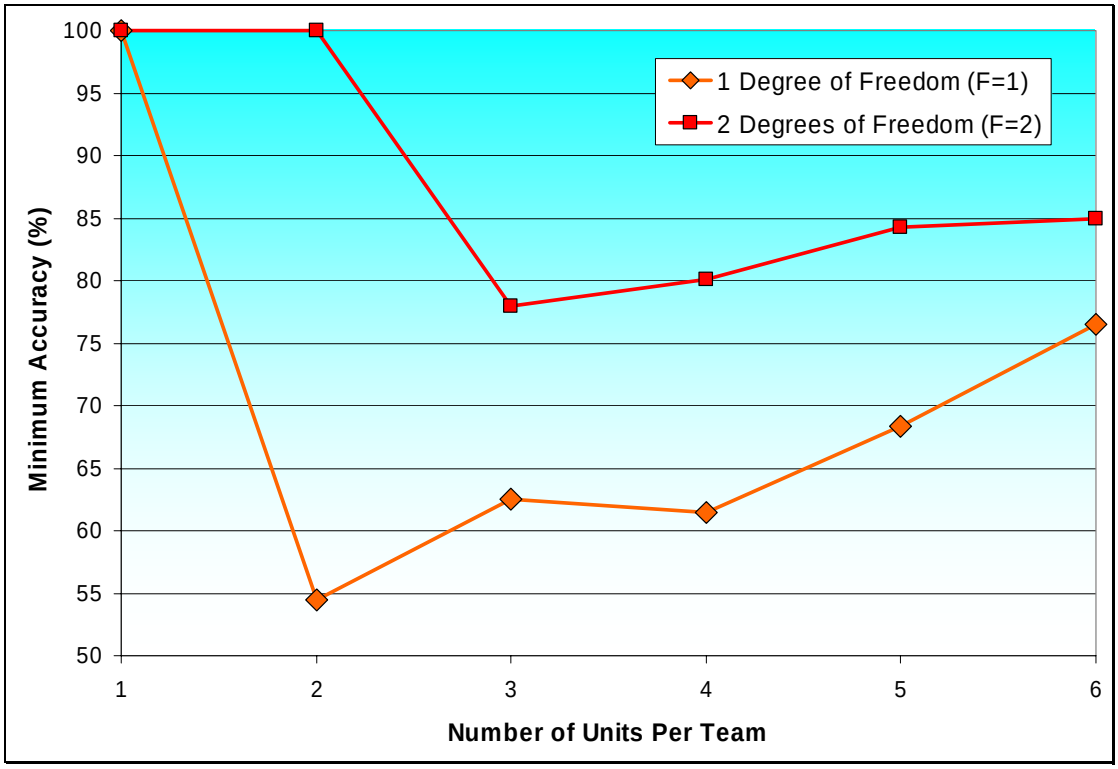

Figure 3.11 – Minimum Accuracy of ULTRA vs. the Number of Units per Team.

First, it should be noted that in the cases in which there is only one unit per side, or when $F = 2$ and there are two units per side, ULTRA corresponds to the exhaustive search. Consequently, ULTRA can never do worse than the optimal target assignment strategy. Second, the worst case performance of the ULTRA algorithm appears to improve asymptotically as the number of units increase. This behavior can be explained in that in cases with a small number of units per side, a target assignment error can have a great effect on the performance of the target assignment strategy. However, in a case with a large number of units per side, the effect of a single target assignment error is mitigated by the larger number of units. This results in less of a performance decrease than in cases considering fewer numbers of units. Third, the difference between a degree of freedom of one and a degree of freedom of two appears to be much smaller especially

at high numbers of units per side. This is in contrast to what was seen in the average accuracy measurements. However, the curves are not smooth so the discrepancy might be a result of random perturbations.

A third set of accuracy measurements to consider is a threshold based performance measure. This measurement describes the probability that ULTRA will arrive at a target assignment strategy that results in a objective function value within a certain percentage of the optimal value. In this experiment we consider four different thresholds. Figure 3.12 illustrates the probability of ULTRA achieving the exact optimal solution as the number of units per side varies.

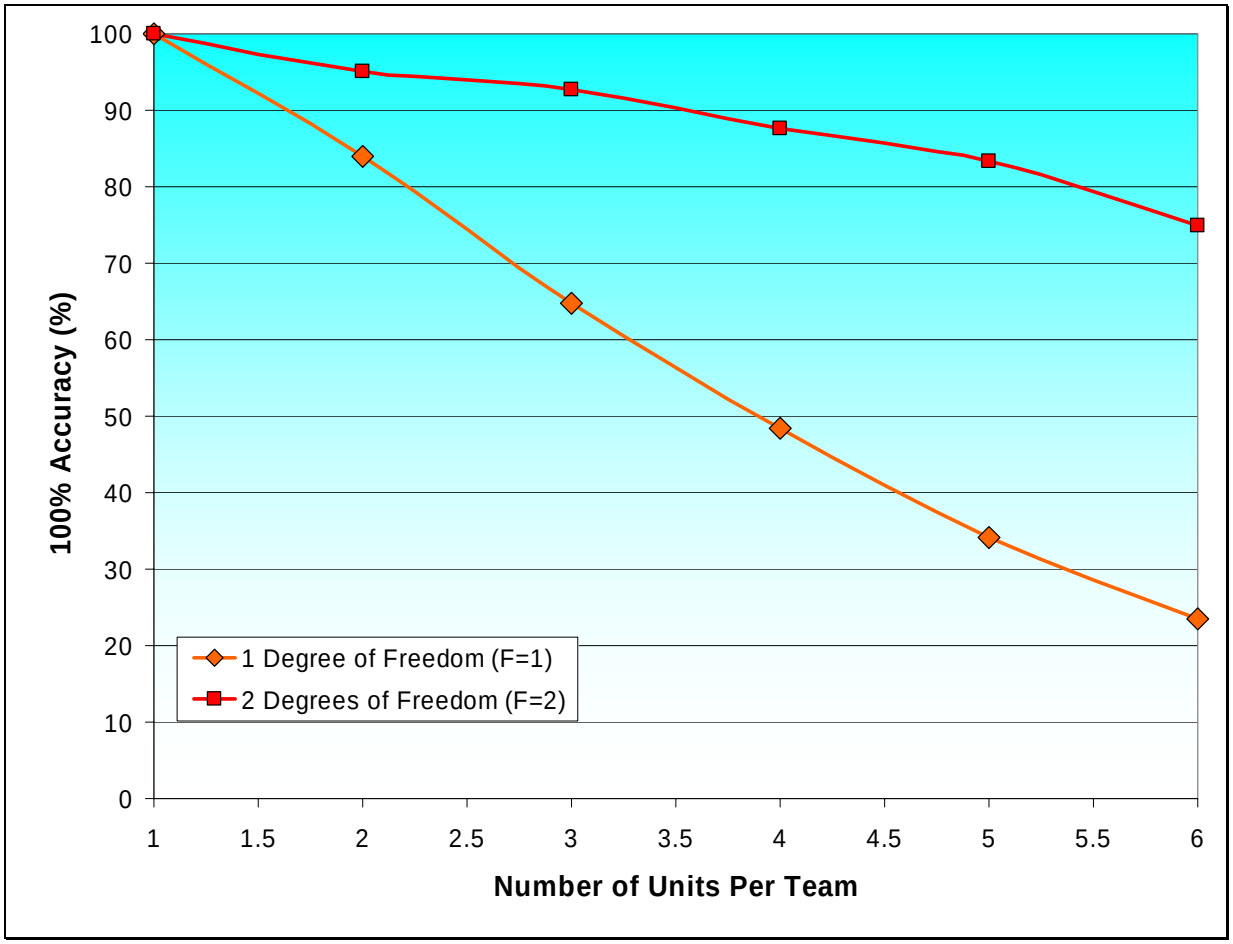

Figure 3.12 – %Chance of ULTRA returning the Optimal Target Assignment Strategy

This shows that with F=1 the probability of ULTRA returning the exact optimal strategy falls quickly to zero. Second, we measured the probability that the ULTRA algorithm returns a target assignment strategy resulting in an objective function value greater than or equal to 99 percent of the optimal value. The results of these measurements are compiled in Figure 3.13. It can be seen that these measurements are similar to those illustrated in Figure 3.12.

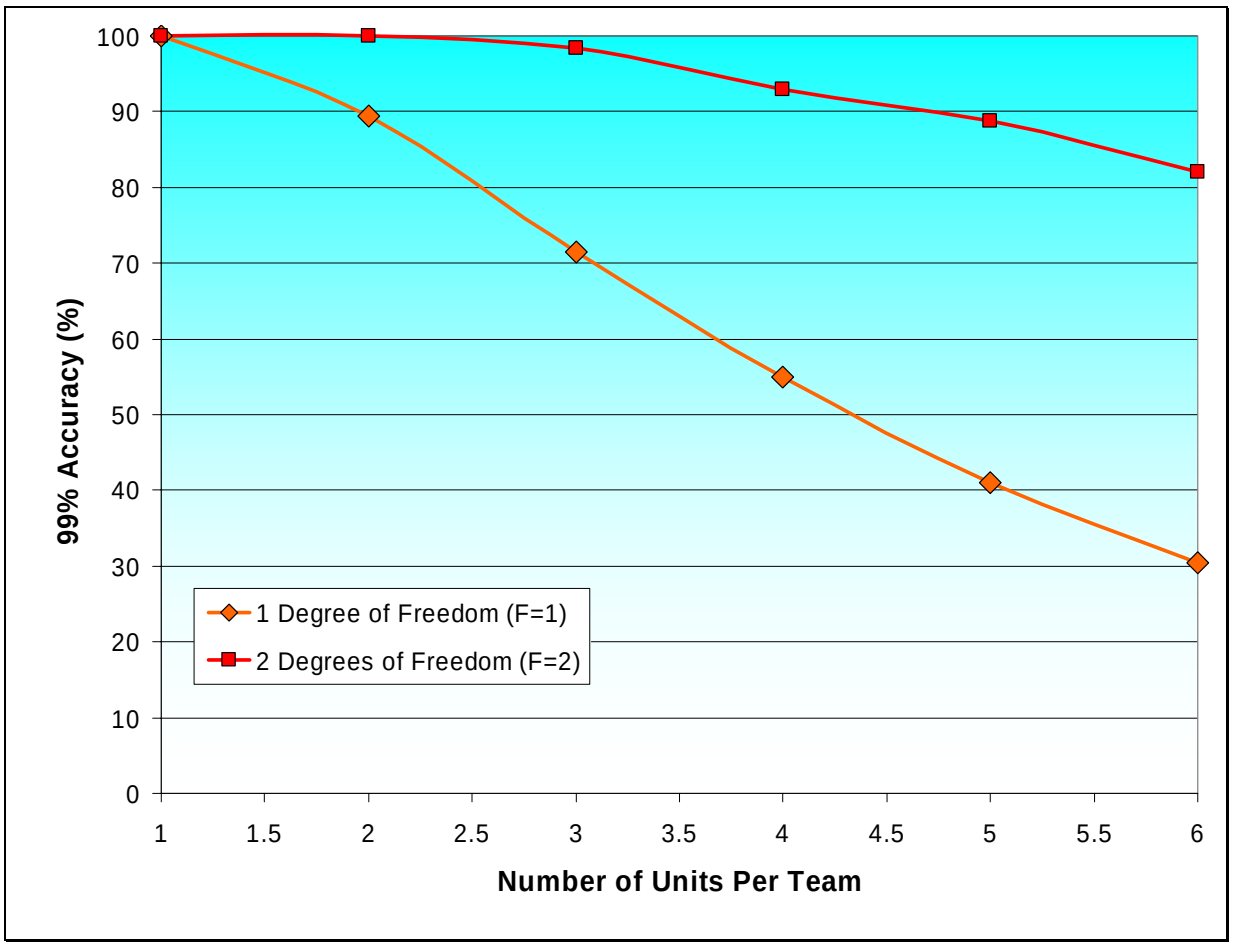

Figure 3.13 – Chance of ULTRA Obtaining a Target assignment Strategy > 99% Optimal

Third, we measured the probability that the ULTRA algorithm obtains a target assignment strategy resulting in an objective function value greater than or equal to 95% of the optimal value. This measurement is shown in Figure 3.14. Unlike the previous two measurements, the curves for both *F=1* and *F=2* clearly appear to converge to a non-zero value. It can be extrapolated that when *F=1*, ULTRA will generate a target assignment strategy that is 95% of the optimal or better roughly 50% of the time for large numbers of units per team. When *F=2* the results are even more definitive. ULTRA will generate a target assignment strategy better than 95% of the optimal approximately 95% of the time for large numbers of units per team.

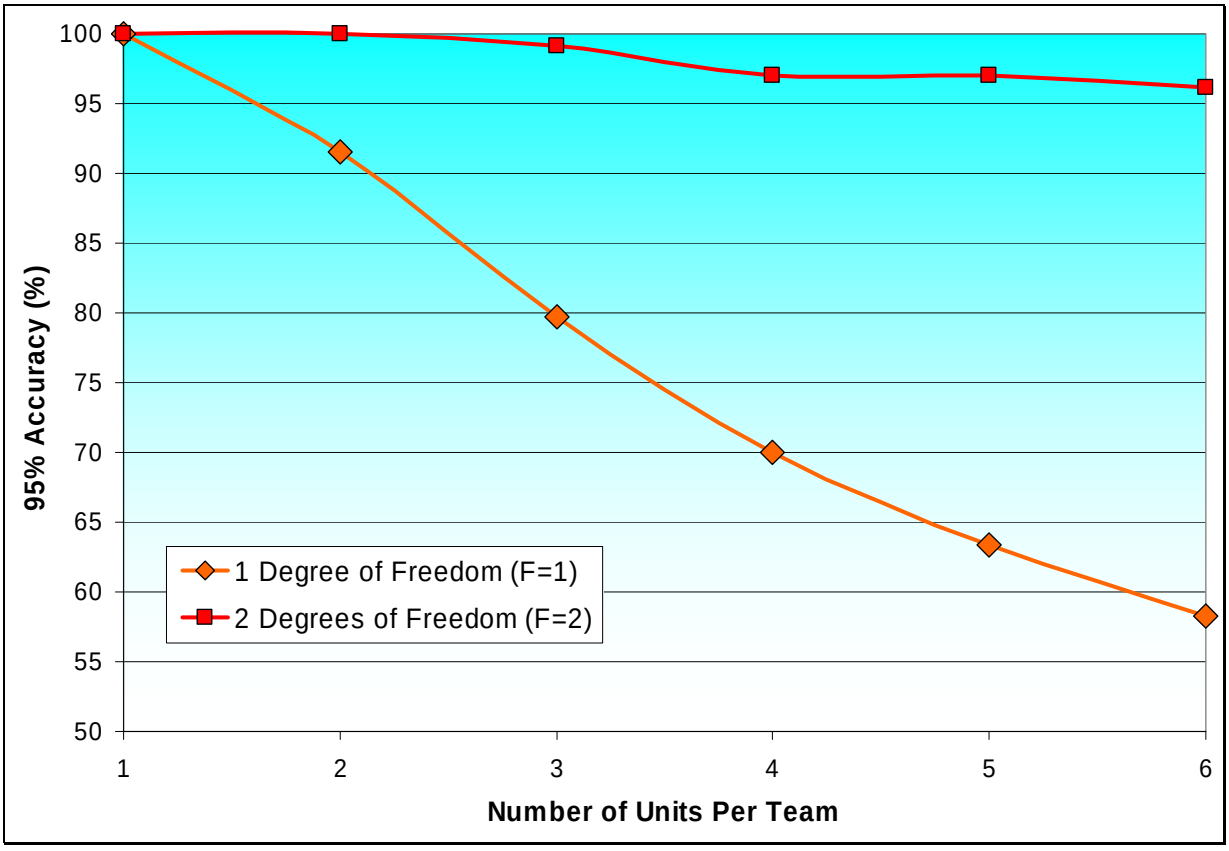

Figure 3.14 – Chance of ULTRA Obtaining a Target assignment Strategy > 95% Optimal

Fourth, in Figure 3.15 we measured the probability that the ULTRA algorithm returns a target assignment strategy resulting in an objective function score better than 90% of the global optimum. Much like the worst case measurements, this measurement shows the peculiar property that when *F=1* the probability of obtaining a target section strategy better than 90% of the optimum increases with higher numbers of units.

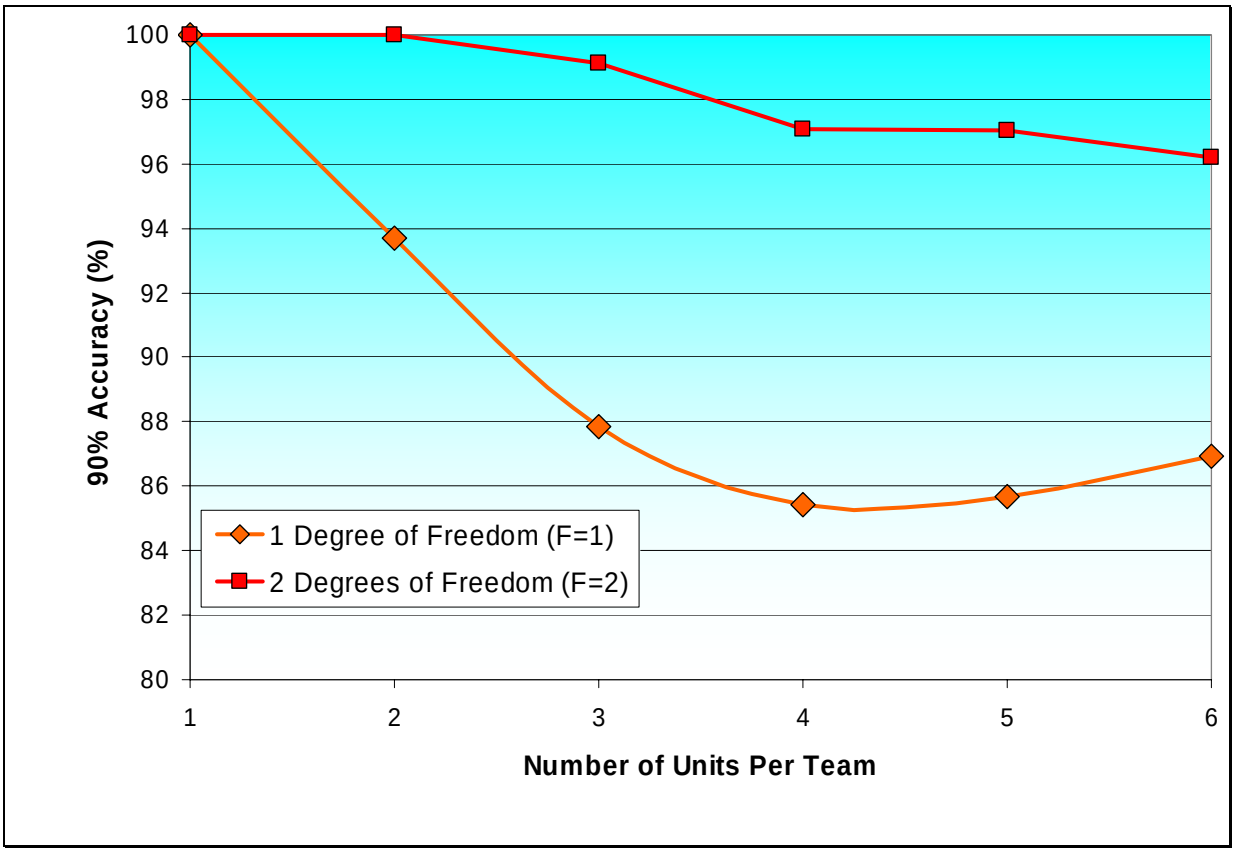

Figure 3.15 – Chance of ULTRA Obtaining a Target assignment Strategy > 90% Optimal

To better illustrate the differences between the four different threshold measurements considered we compiled them in a common Figure. Figure 3.15 contains a compilation of the threshold measurements taken in Experiment I when *F=1*.

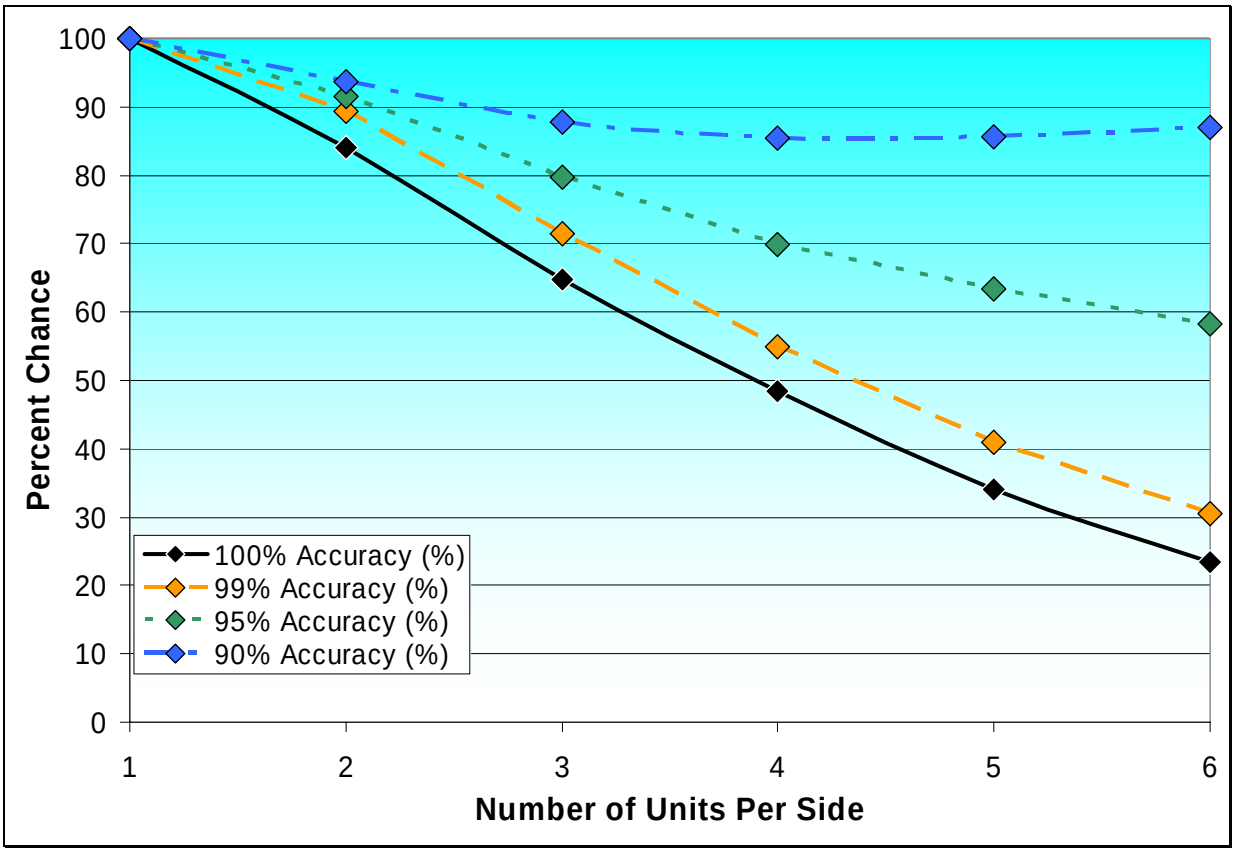

Figure 3.16 – Comparison of Threshold Measurements Considered when *F=1*

Run time requirements are another important factor to consider when measuring the performance of an algorithm. The fourth performance measurement that we considered in Experiment I is the worst case run time of the ULTRA algorithm. This data is given in Figure 3.17.

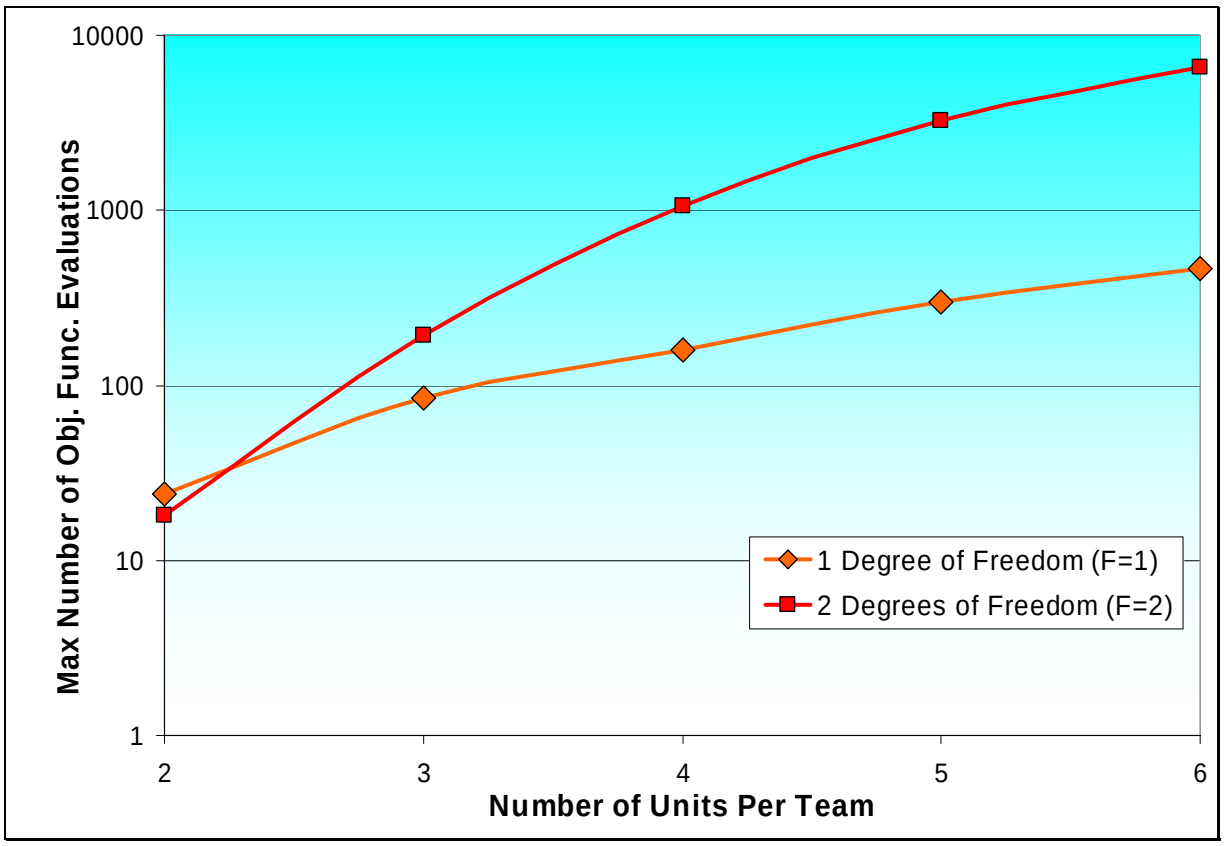

Figure 3.17 – Maximum Number of Objective Function Evaluations

We are able to draw several conclusions from Experiment I. First, we have established that for large numbers of units on each side, on average the ULTRA algorithm is 94% accurate for *F=1* and 98% accurate when the degree of freedom coefficient is set to two. Second, when *F=1*, ULTRA is seemingly guaranteed to generate a target assignment strategy resulting in an objective function value better than 75% of the optimum. Third, for large numbers of units per team, when *F=1* the ULTRA algorithm will arrive at a target assignment that is better than 90% of the optimum approximately 90% of the time. Additionally, when *F=2* ULTRA will generate a target assignment strategy better than 95% of the optimum approximately 95% of the time.

3.4.2 Experiment II

In LSNS type algorithms, the initial target assignment strategy can affect the overall performance of the algorithm. In the second Experiment we examined the effect of the initial target assignment strategy on the performance of the ULTRA algorithm. Here we compared the performance of the three previously mentioned target assignment strategies, the zero target assignment strategy, the random target assignment strategy, and the unit greedy target assignment strategy. Setting the degree of freedom coefficient to one, we evaluated the ULTRA algorithm for each of the three initial strategies. For each initial condition, as in Experiment I, we defined the number of units per side such that $N_A = N_B \in \{1,2,3,4,5,6\}$. Additionally, for each scenario considered, the model was simulated 25,000 times, with each instance having randomly generated probability of kill values such that $P_i^B = 1 \; \forall i \in \{1,2,\ldots,N_B\}$.

The first aggregate measurement considered is average accuracy. Figure 3.18 shows the average percentage of the optimal objective function value returned for each of the three initial conditions. Clearly, on average the unit greedy initial condition yields the most optimal target assignment strategies. In fact, the unit greedy initial condition finds the exact optimal target assignment strategy in the case of two units per team. The random target assignment strategy performs the worst of the three, about 4% worse than the zero target assignment strategy and about 7% worse than the unit greedy approach.

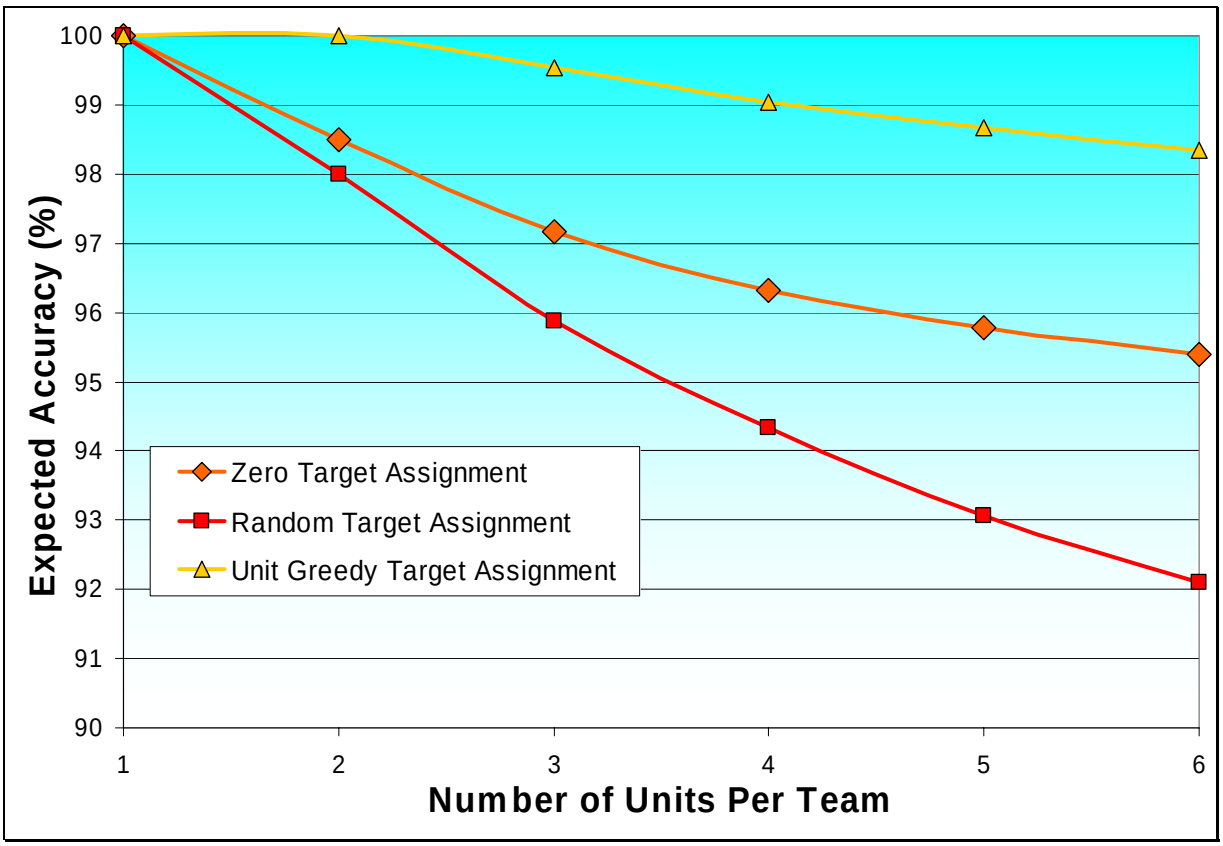

Figure 3.18 – Expected Accuracy for Various Initial Strategies

Second, we compared the worst case accuracy of the three initial conditions. This measurement is illustrated in Figure 3.19. Again, the best results are obtained with the ULTRA algorithm when the unit greedy target assignment is used as the initial strategy. When this initial strategy is used, ULTRA generates target assignment strategies better than 84% of optimal. For scenarios with larger numbers of units per team, the zero initial target assignment strategy is a close second, with nearly 80% minimum accuracy. Finally, the random target assignment strategy is again the worst performing of the three.

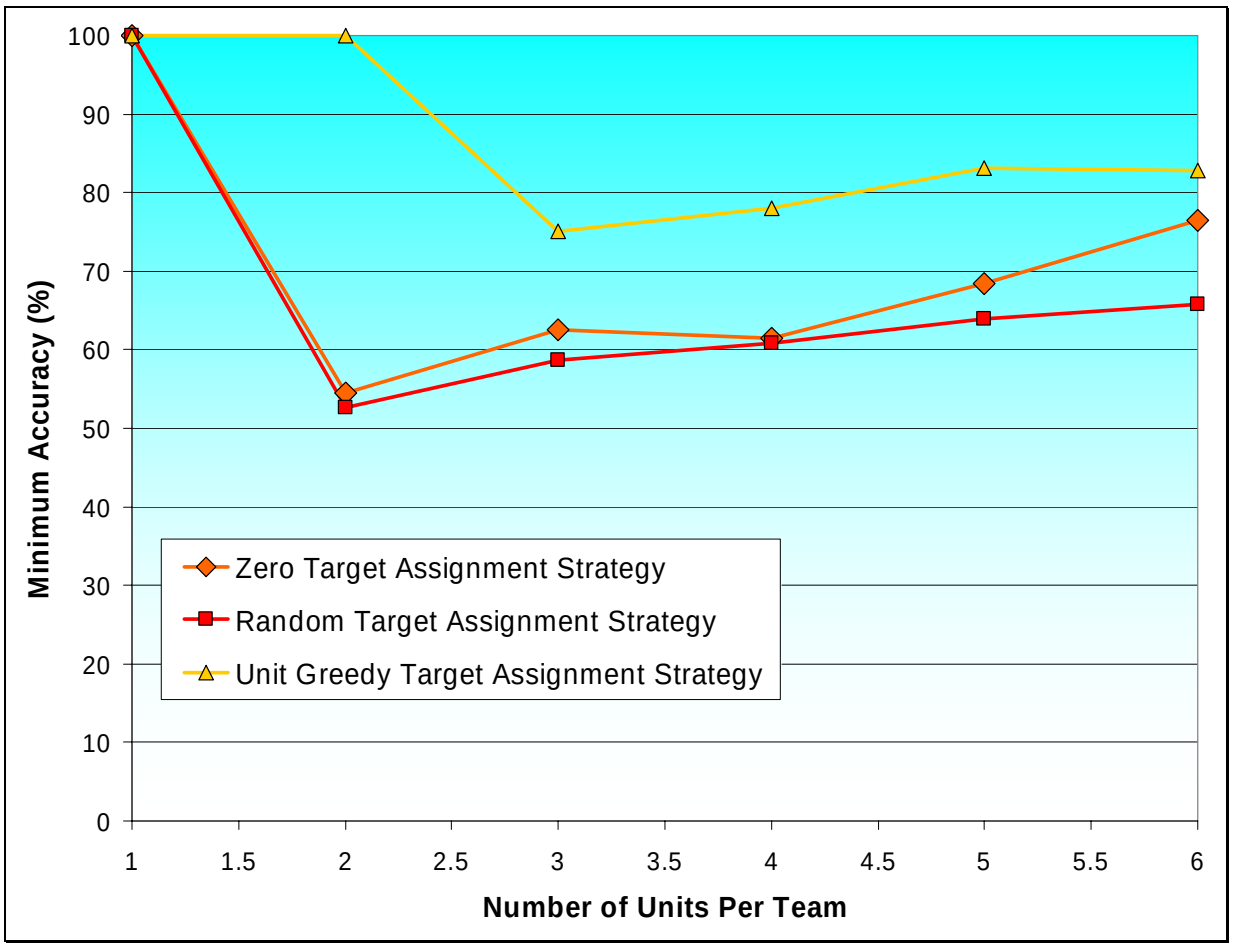

Figure 3.19 – Worst Case Accuracy for Various Initial Strategies

Third we considered how the average run time requirements change as a function of the initial target assignment strategies. A plot comparing the average number of objective function evaluations required for ULTRA to converge for each of the three initial strategies is shown in Figure 3.20. These three curves appear to have a definite symmetry. On average, the quickest of the three initial strategies is the unit greedy approach. When the random approach is used the algorithm takes approximately 60% longer to terminate than when using the unit greedy

approach. Finally, the zero assignment approach takes the longest, requiring 50% more objective function evaluations than the random and 140% more than the unit greedy approach.

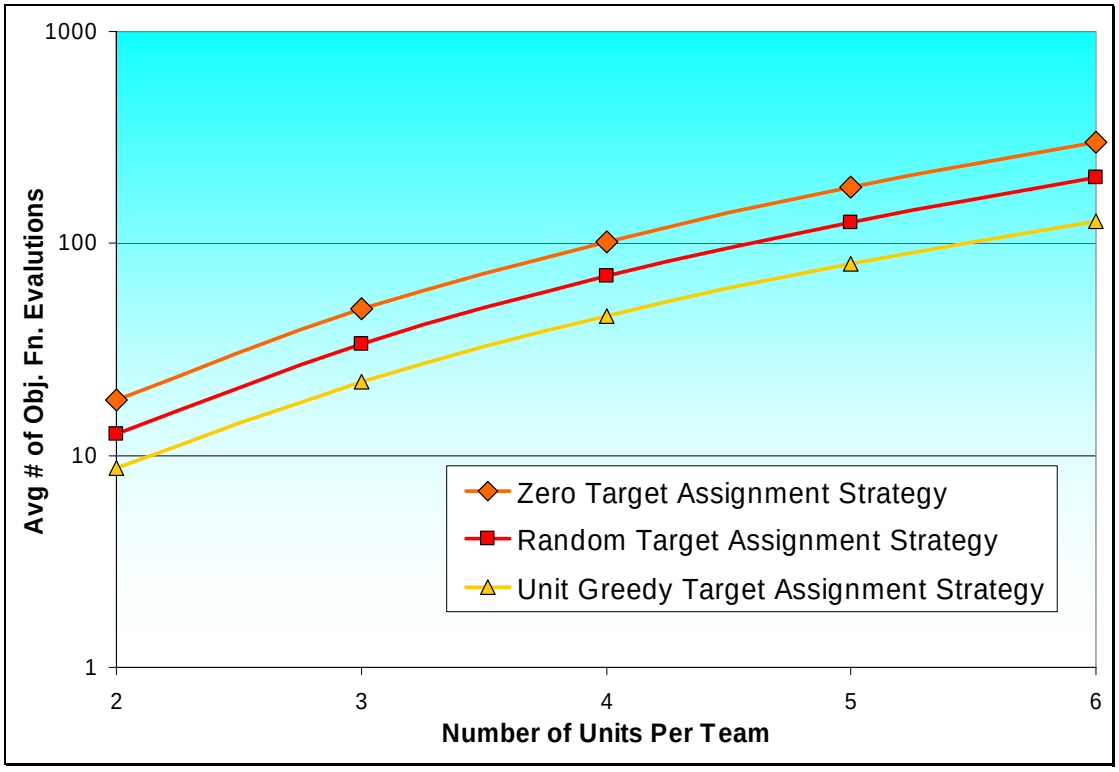

Figure 3.20 – Avg # of Obj Fn Evaluations for Various Initial Strategies

A fourth consideration is the maximum run time requirements of the various initial conditions. Figure 3.21 plots the maximum number of objective functions required for ULTRA to terminate over 25,000 random instances versus the number of units per team for the three initial strategies. Two important points can be made from this measurement. First, there appears to be no difference in the maximum number of objective function evaluations when ULTRA uses the random compared to the greedy target assignments as initial strategies. This is an important

point because the random target assignment strategy encompasses all target assignment strategies. This implies that no other initial assignment strategy generates a higher maximum run time than the zero initial strategy. Second, the maximum number of objective function evaluations required to converge when ULTRA uses the unit greedy target assignment as the initial strategy is approximately equal to the average number of objective function evaluations required by the zero initial assignment strategy.

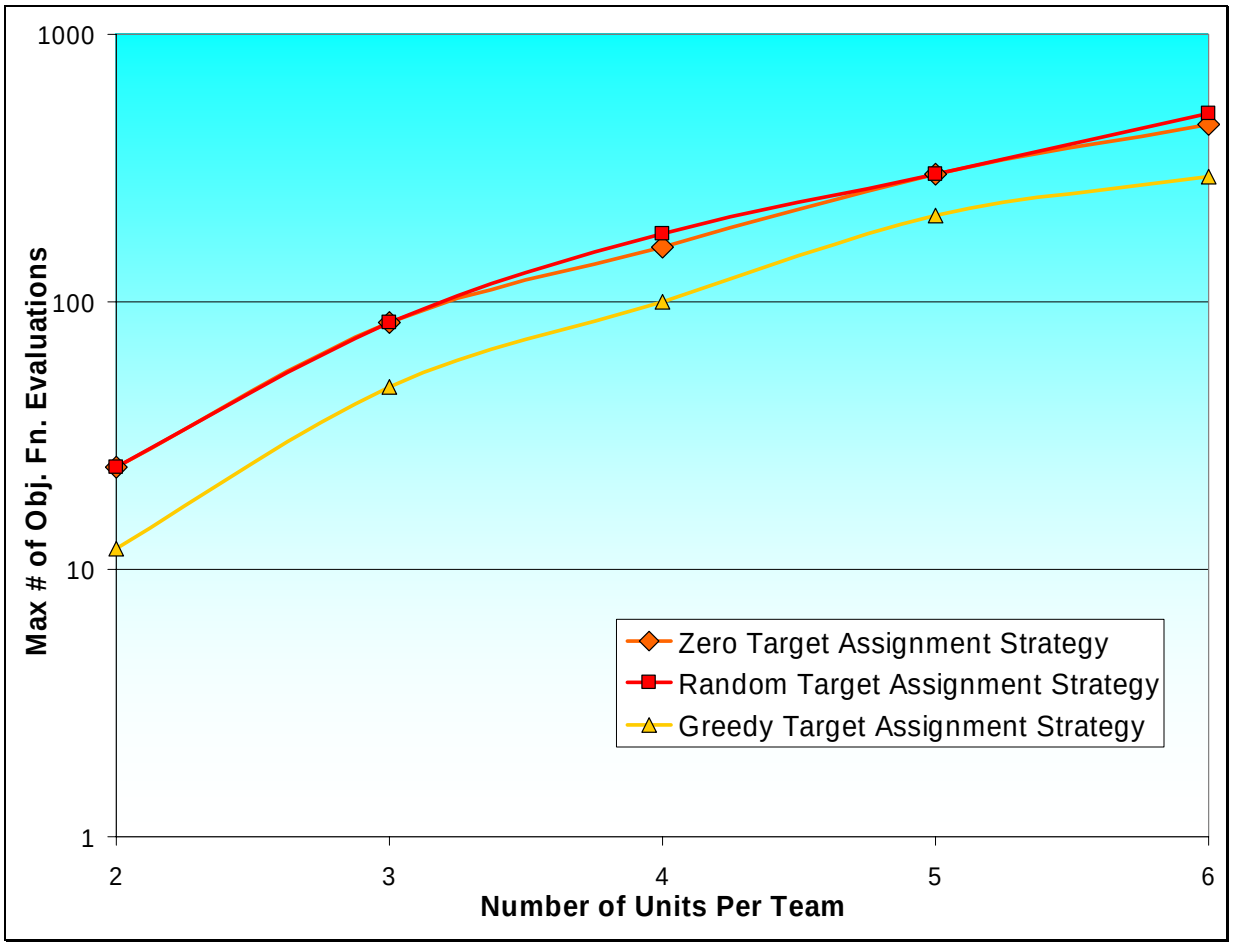

Figure 3.21 – Maximum # of Obj Fn Evaluations for Various Initial Strategies

Fifth, we measured the minimum number of objective function evaluations required for ULTRA to converge as a function of the initial strategy and the number of units per side. This measurement is illustrated in Figure 3.22. Again, two points can be made about this measurement. First, the ULTRA algorithm achieves the best possible convergence rates when either the random or the unit greedy target assignments are used as initial conditions. This is expected as it is possible for either initial strategy to be optimal in its given neighborhood.

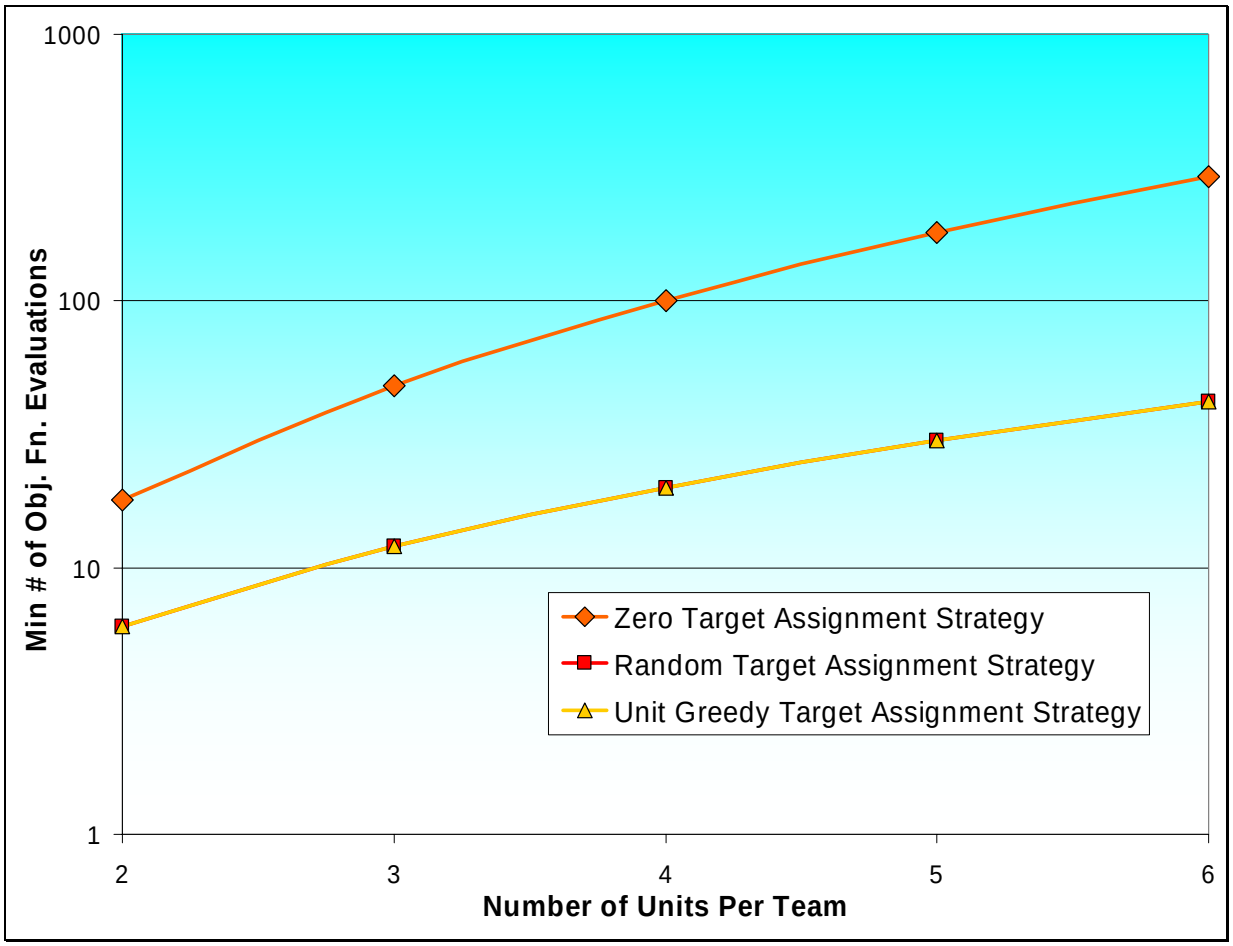

Figure 3.22 – Maximum # of Obj Fn Evaluations for Various Initial Strategies

Second, when ULTRA uses the zero target assignment as the initial strategy, the quickest instance of convergence appears to be only slightly faster than the average for the same initial strategy.

Experiment II compared the performance of three different types of target assignments as they are used as the initial strategy of the ULTRA algorithm. From the measurements taken, we are able to make three conclusions. First, the unit greedy target assignment strategy is the best target assignment strategy of the three in both accuracy and run time requirements. It presents a significant, roughly 60%, improvement over the zero strategy in average number of objective function evaluations and is roughly 4% more accurate. Second, the zero target assignment strategy is the worst possible initial strategy in terms of maximum number of objective function evaluations. Third, although the maximum number of objective function evaluations is much higher, there is very little difference between the minimum and the average number of objective functions required for convergence when ULTRA uses the zero initial target assignment strategy.

3.4.3 Experiment III

In the previous two experiments we have only considered cases in which Team A and Team B both have the same number of units. In general, this is not the case. In Experiment III we measured the performance of the ULTRA algorithm as the number of units on each team varies independently. Here we assumed that ULTRA used the either the zero or the unit greedy initial target assignment strategy. As in the previous two experiments, the model was simulated 25,000 times for each scenario considered, each instance having randomly generated probability of kill values such that $P_i^B = 1 \; \forall i \in \{1, 2, \ldots, N_B\}$.

First, we examined two cases. The first, in which the number of units on Team A varies from one to 6 while the number of units on Team B remains constant at four and the second in which the number of units on Team B varies from one to 6 while the number of units on Team A remains constant. We evaluated each of these scenarios using both the unit greedy assignment and the zero assignment as the initial strategy using four metrics. In the first metric, we measured how the average accuracy was affected by non-uniformly varying the number of units per team. The resulting data can be seen in Figure 3.23.

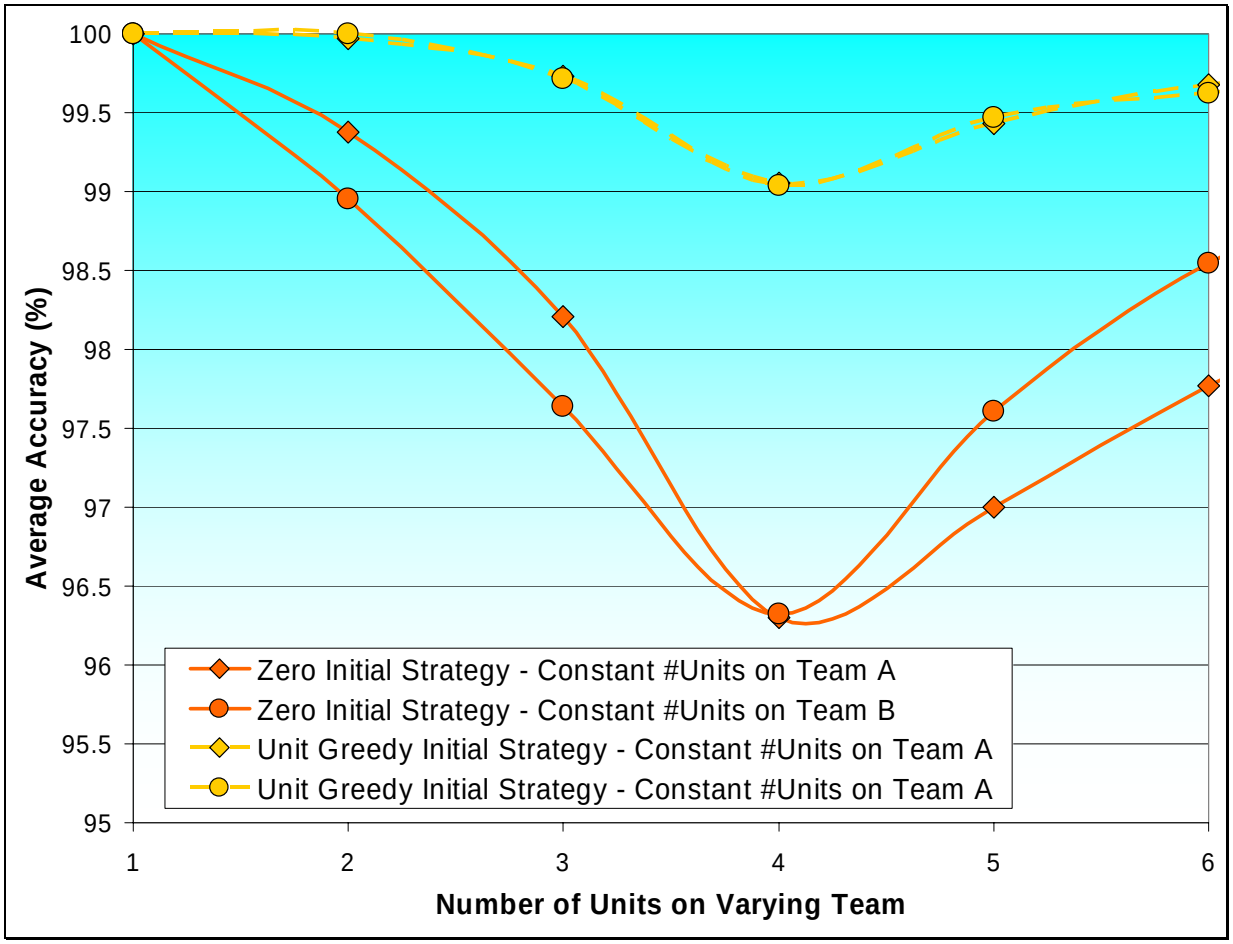

Figure 3.23 – Average Accuracy of ULTRA with Dissimilar Teams

Several interesting concepts are illustrated in Figure 3.23. On one hand, the lowest average accuracy occurred when Teams A and B had an equal number of units, the case when both were composed of four units. ULTRA appears to generate better strategies for situations involving teams of dissimilar numbers of units than for more balance situations. On the other, when using the unit greedy target assignment strategy, the average accuracy of the strategy ULTRA calculates for Team A is approximately the same when there are many units on Team A and few on Team B as in the case when there are few units on Team A and many on Team B. This symmetry is not found when the zero target assignment strategy is used as the initial strategy. For example, the when the zero initial strategy is employed, on average ULTRA generates a better Strategy for Team A when $N_A = 6$ and $N_B = 4$ then when $N_A = 4$ and $N_B = 6$.

In the second metric we explored the manner in which having teams of dissimilar size effects the threshold performance of ULTRA. In particular we examined the probability that ULTRA returns an optimal strategy. These measurements are shown in Figure 3.24. From Figure 3.24 we can conclude that ULTRA often yields an optimal target assignment strategy when there are few units on Team A and many units on Team B. In contrast, ULTRA almost never yields an optimal target assignment strategy when there are many units on Team A and few units on Team B. This makes sense as when there are few units on Team A and many on Team B each unit on Team A will often be assigned to the Unit on Team B it is most suited to attack. On the other, when there are many units on Team A and few units on Team B it can be very difficult to coordinate an optimal attack.

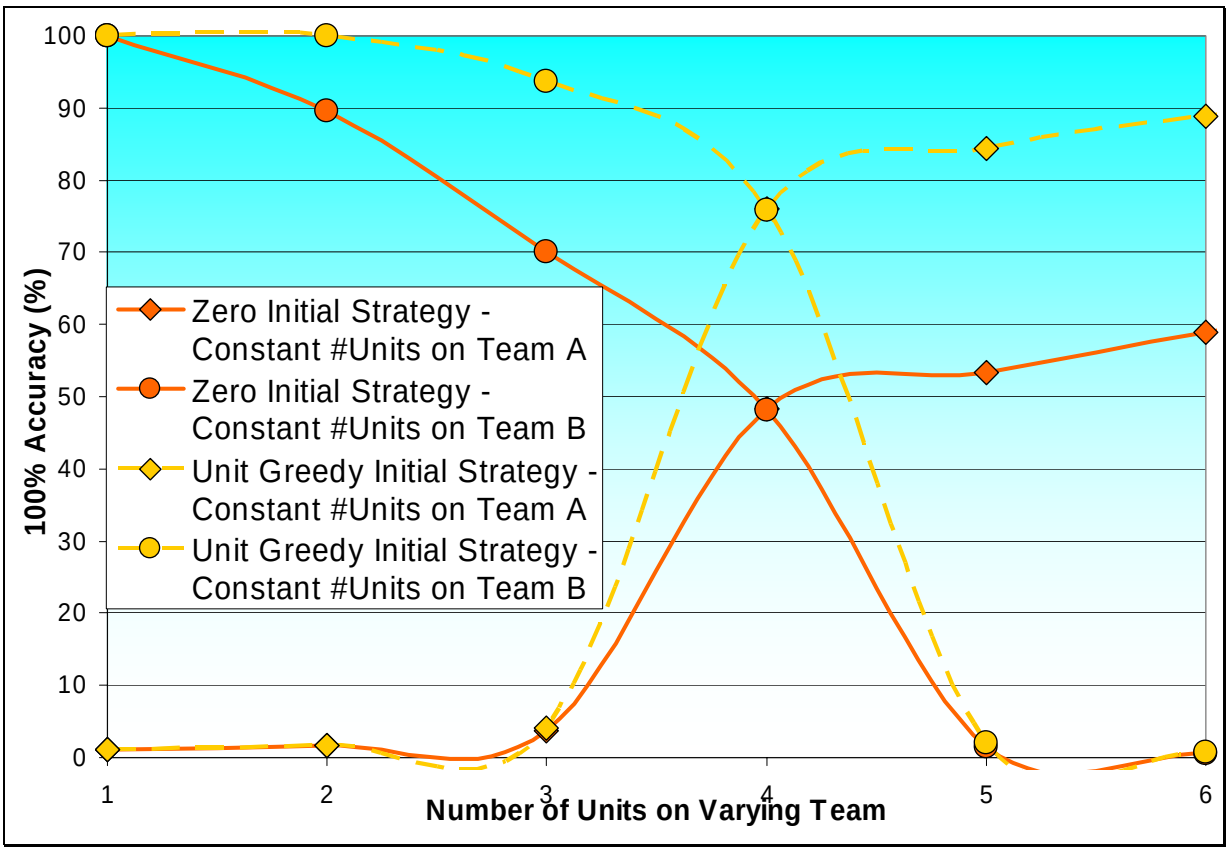

Figure 3.24 – Chance of ULTRA obtaining Optimal Strategy with Dissimilar Teams

The third metric used to examine the effects of teams of dissimilar numbers of units is the run time requirements of ULTRA. In particular, we examined the average number of objective function evaluations required for ULTRA to converge. This is shown in Figure 3.25. Here we see that the number of units on Team A has a larger effect on the time required for ULTRA to calculate a target assignment strategy for Team A than the number of units on Team B.

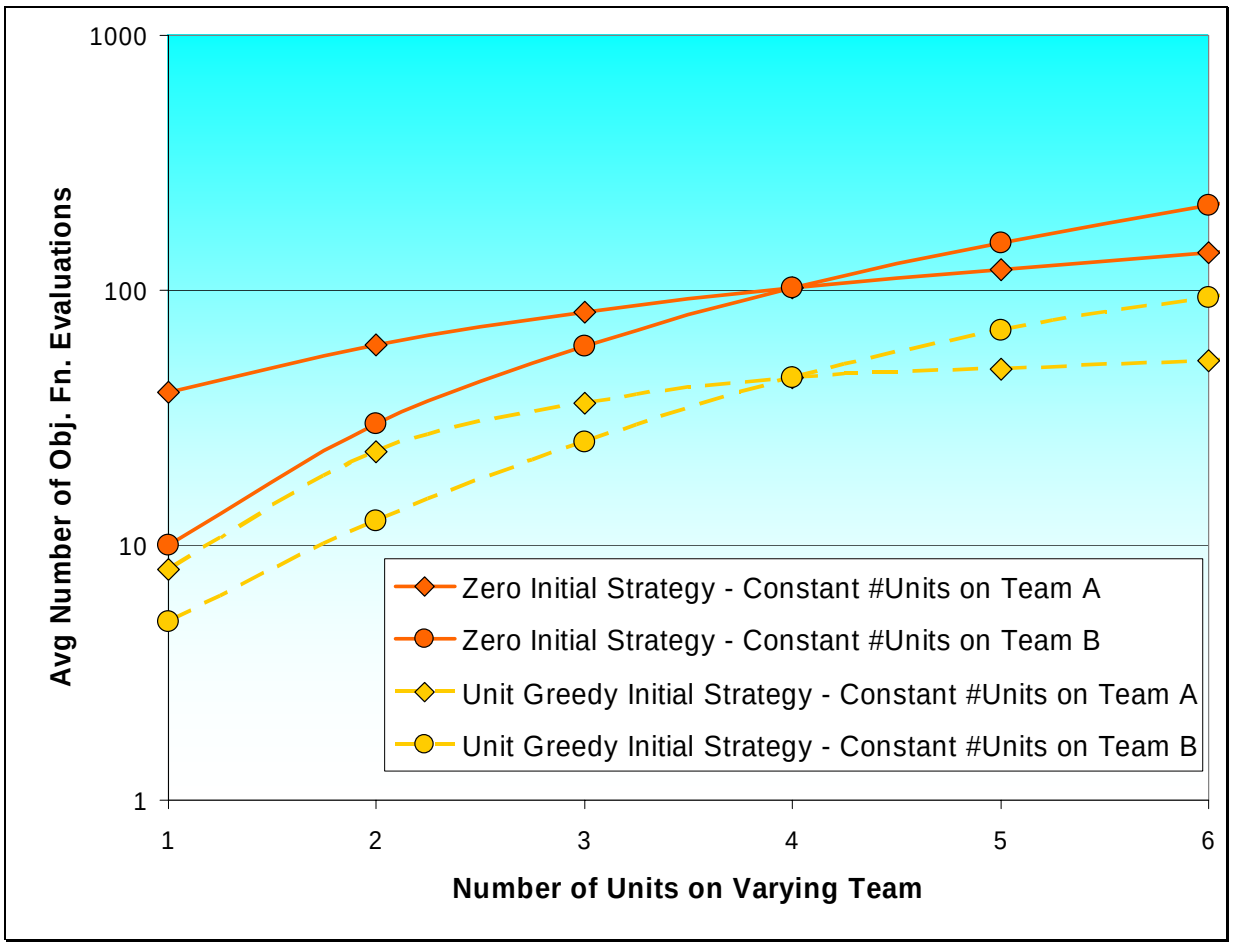

Figure 3.25 – Avg. Number of Objective Function Evaluations Assuming Dissimilar Teams

In the second part of Experiment III, instead of fixing the number of units on one team and varying the number of units on the other we examined all combination of number of units per team when $N_A \in \{1,2,3,4,5,6\}$ and $N_B \in \{1,2,3,4,5,6\}$. Every scenario we measured the average accuracy of the ULTRA algorithm for each of the three initial target assignment strategies presented earlier over 25,000 iterations. Figure 3.26 illustrates the result of this simulation. From these plots, we can conclude that the conclusions drawn previously in

Experiment III are applicable in a more general sense. These conclusions seem to apply to all combinations of units on each side rather that the specific case when one team has four units.

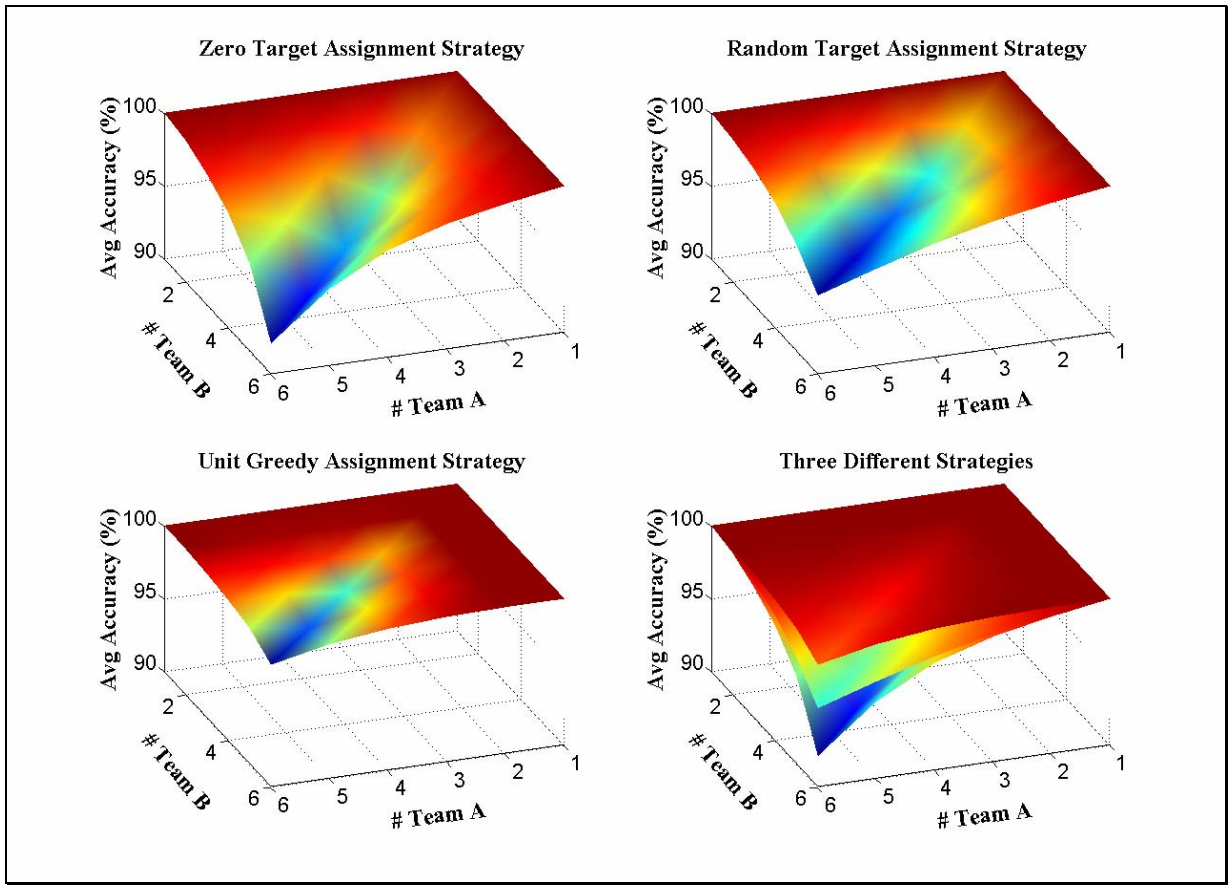

Figure 3.26 – Surface Plots of Avg Accuracy versus # of Units on Team A and B

4.0 FINDING A NASH SOLUTION IN MT-DWTA PROBLEMS

In the previous chapter we discussed a special case of the multi-team dynamic weapon target assignment problem. We presented a method for quickly generating a target assignment strategy when the strategies for all other teams are known a-priori. Knowledge of this sort is seldom available in competitive target assignment scenarios. Competitive target assignment scenarios typically contain some form of uncertainty about the intended target assignment strategy of a team's opponents instead of direct knowledge about another teams intended strategy. Teams seldom know more than estimates of their opponents' objective functions and strategies. This uncertainty makes it difficult to justify particular solution concepts. On one hand the additional knowledge of an opponent's intent can play a critical role in outcome of the scenario and should not be ignored. Conversely, there is often a desire to employ a naïve approach, ignoring the consequences of an opponent's possible actions to quickly generate a target assignment strategy. While the Nash equilibrium is an important solution concept when dealing with this type of uncertainty, it is computationally inefficient to calculate using standard approaches. There is also no guarantee that a particular scenario has a Nash solution. Any real world implementation using a Nash based approach must be computationally feasible, robust and must generate strategies that are statistically better than naïve approaches.

Perhaps the largest determent to applying the MT-DWTA to model real world applications is the computational complexity resulting from finding Nash equilibrium strategies using

standard techniques. The most common technique for finding such a strategy is through the use of a game matrix. As we showed in Chapter 2, a game matrix is first formed by assigning an axis to each player in the game. Each axis contains a single index for every possible strategy that the corresponding team may employ. This means that each entry in the game matrix represents a single combination of strategies from each team in the game and that the matrix contains every possible combination of strategies. To fill the game matrix, the cost functions from each team are evaluated and the results are stored for each entry in the game matrix. As a result this game matrix becomes computationally intractable even for small problems. Forming a game matrix to solve the MT-DWTA requires too many computations and too much memory to fill and store the game matrix entries.

To solve the problem of computational feasibility we will examine the standard action reaction Nash equilibrium search. This search enables a game theoretic problem to be solved using standard optimization technique. We also examine games which do not have a Nash solution. To account for these cases we define a more robust game theoretic solution, extending the basic definition of a Nash solution to include the idea of fairness. We will show that by incorporating this idea, our algorithm can robustly handle sufficiently large scenarios within an acceptable amount of time even though some scenarios may not have a Nash equilibrium point. The last topic covered in this chapter is the performance of the Nash strategies when compared to other non game theoretic or naïve strategies. We will examine three other target assignment strategies, the random target assignment, the unit greedy target assignment and the team optimal target assignment. We show that although the Nash target assignment strategy is not optimal in the traditional sense, it performs better than any of the other strategies, regardless of the target assignment strategy of its adversary.

4.1 APPLYING THE ACTION - REACTION NASH SEARCH TO THE MT-DWTA

Any algorithm that uses game theoretic solution concepts to solve the MT-DWTA must solve two distinct problems. The algorithm must be fast, generating target assignment strategies quickly as the size of the problems are usually quite large. Similarly, the algorithm must also make efficient use of memory. One common way to efficiently find the Nash strategy is to use the standard action reaction search. This search bears some similarity to certain decent type searches in which the search moves to a more optimal point along the optimal direction.

To illustrate how the action - reaction search functions, consider a two player game with controls $u_{\text{Player 1}}$ and $u_{\text{Player 2}}$ and objective functions $J_{\text{Player 1}}\left(u_{\text{Player 1}}, u_{\text{Player 2}}\right)$ and $J_{\text{Player 2}}\left(u_{\text{Player 1}}, u_{\text{Player 2}}\right)$. To begin the action - reaction search we first assign some strategy $u^0_{\text{Player 1}}$ to Player A. Player 2's optimal response $u^0_{\text{Player 2}}$ is then calculated according to the following maximization:

$$J_{\text{Player 2}}\left(u^0_{\text{Player 1}}, u^0_{\text{Player 2}}\right) \geq J_{\text{Player 2}}\left(u^0_{\text{Player 1}}, u_{\text{Player 2}}\right) \quad \forall u_{\text{Player 2}}. \tag{4.1}$$

Following this, Player 1 calculates the optimal response to Player 2's strategy[3] $u^0_{\text{Player 2}}$ using a similar optimization:

$$J_{\text{Player 1}}\left(u^1_{\text{Player 1}}, u^0_{\text{Player 2}}\right) \geq J_{\text{Player 1}}\left(u_{\text{Player 1}}, u^0_{\text{Player 2}}\right) \quad \forall u_{\text{Player 1}}. \tag{4.2}$$

These expressions given in (4.1) and (4.2) can be generalized to the following representation of the action - reaction search:

[3] It should be noted that players do not actually announce their strategies in the game. The concept of optimal responses to a given strategy is a computational tool used solely by the decision maker of a single player.

$$J_{\text{Player 1}}\left(u^{\iota}_{\text{Player 1}}, u^{\iota-1}_{\text{Player 2}}\right) \geq J_{\text{Player 1}}\left(u_{\text{Player 1}}, u^{\iota-1}_{\text{Player 2}}\right) \quad \forall u_{\text{Player 1}}$$
$$J_{\text{Player 2}}\left(u^{\iota}_{\text{Player 1}}, u^{\iota}_{\text{Player 2}}\right) \geq J_{\text{Player 2}}\left(u^{\iota}_{\text{Player 1}}, u_{\text{Player 2}}\right) \quad \forall u_{\text{Player 2}} \tag{4.3}$$

This process iterates until on of three possible conditions is reached:

1. $u^{\iota}_{\text{Player 1}} = u^{\iota-1}_{\text{Player 1}}$ or $u^{\iota}_{\text{Player 2}} = u^{\iota-1}_{\text{Player 2}}$ (4.4)

The first condition represents the necessary conditions for a Nash equilibrium. To illustrate this point, note that $u^{\iota}_{\text{Player 1}} = u^{\iota-1}_{\text{Player 1}}$ implies that $u^{\iota}_{\text{Player 2}} = u^{\iota-1}_{\text{Player 2}}$ and visa versa. Since $u^{\iota-1}_{\text{Player 2}}$ is an optimal response to $u^{\iota-1}_{\text{Player 1}}$ and $u^{\iota}_{\text{Player 1}} = u^{\iota-1}_{\text{Player 1}}$ we can say that $u^{\iota-1}_{\text{Player 2}}$ is also an optimal response to $u^{\iota}_{\text{Player 2}}$. This means that $u^{\iota-1}_{\text{Player 2}}$ is a possible solution to (4.3) at iteration ι. Also recall the definition of the Nash equilibrium given in (2.1). Applying this definition to the current example, we find that a pair of strategies $\left(u^{N}_{\text{Player 1}}, u^{N}_{\text{Player 2}}\right)$ is a Nash pair of strategies if:

$$J_{\text{Player 1}}\left(u^{N}_{\text{Player 1}}, u^{N}_{\text{Player 2}}\right) \geq J_{\text{Player 1}}\left(u_{\text{Player 1}}, u^{N}_{\text{Player 2}}\right) \quad \forall u_{\text{Player 1}}$$
$$J_{\text{Player 2}}\left(u^{N}_{\text{Player 1}}, u^{N}_{\text{Player 2}}\right) \geq J_{\text{Player 2}}\left(u^{N}_{\text{Player 1}}, u_{\text{Player 2}}\right) \quad \forall u_{\text{Player 2}}.$$

This is equivalent to (4.3) when $u^{\iota}_{\text{Player 1}}$ and $u^{\iota}_{\text{Player 2}}$ are substituted in for $u^{\iota-1}_{\text{Player 1}}$ and $u^{\iota-1}_{\text{Player 2}}$.

2. $u^{\iota}_{\text{Player 1}} = u^{\iota-\varphi}_{\text{Player 1}}$ or $u^{\iota}_{\text{Player 2}} = u^{\iota-\varphi}_{\text{Player 2}}$ where $\varphi \geq 2$ (4.5)

The second condition is difficult to classify. It is a similar condition to that often seen in decent type algorithms where numerical errors cause the algorithm to continually cycle past the optimal solution. Such solutions are easily justifiable in

standard optimization problems but not in Game Theoretic problems. This is because it is much more difficult to evaluate "closeness" to a Nash solution in the same way as solution can be close to the optimal solution in a general optimization problem. In a standard optimization problem, the solution space can usually be assumed to be convex around an optimum so that solutions that are similar to the optimal solution can be assumed to be near optimal; a property not shared by the Nash equilibrium

3. $\iota \geq \mathrm{I}$ where I is the maximum number of iterations (4.6)

The third condition represents the case of non-convergence. While we find that the action - reaction search typically converges in a few iterations in MT-DWTA type games and we know that it is guaranteed to converge to one of the previous 2 conditions, the action - reaction search can theoretically require a number of iterations equal to the minimum number of strategies available to each team.

For example, consider the worst case scenario where there are S strategies available to each team where $u_{\text{Player 1}}$ and $U_{\text{Player 2}}$ are the strategies selected by Player 1 and Player 2 respectively. These strategies are labeled $1,2,\ldots,S$ such that $u_{\text{Player 1}} \in \{1,2,\ldots,S\}$ and $u_{\text{Player 2}}$ and ordered such that Player 1's optimal reaction, $u^*_{\text{Player 1}}\left(u_{\text{Player 2}}\right)$ to a strategy $u_{\text{Player 2}}$ of Player 2 can be expressed:

$$u^*_{\text{Player 1}}\left(u_{\text{Player 2}}\right) = \begin{cases} u_{\text{Player 2}} + 1 & u_{\text{Player 2}} < S \\ 1 & u_{\text{Player 2}} = S \end{cases}$$

and likewise Player 2's optimal reaction $s_2^*\left(s_1\right)$ to a strategy s_1 of Player 1 can be expressed:

$$u^{*}_{\text{Player 2}}\left(u_{\text{Player 1}}\right)=\begin{cases} u_{\text{Player 1}}+1 & u_{\text{Player 1}}<S \\ 1 & u_{\text{Player 1}}=S \end{cases}.$$

In this example it is clear that the action - reaction search will require S iterations before either team repeats a given strategy.

It should be noted that while such an example is theoretically possible, it means little to the action - reaction search. Not only is there no Nash equilibrium in such examples, it is not possible for a team to predict its opponent's intended strategy. In these types of problems a player should either employ a non-game theoretic strategy or a mixed strategy based on randomly selecting a strategy from a weighted set of strategies.

To illustrate how the action reaction search works on a game, recall the simple game defined by the game matrix given in Table 2.2 and reprinted as follows:

Table 4.1 – Sample Game Matrix with a Single Nash Solution

	$u_{\text{Player 2}}=1$	$u_{\text{Player 2}}=2$	$u_{\text{Player 2}}=3$	$u_{\text{Player 2}}=4$
$u_{\text{Player 1}}=1$	$J_A^{*}(1,1)=5$ $J_B(1,1)=15$	$J_A(1,2)=6$ $J_B(1,2)=4$	$J_A(1,3)=14$ $J_B(1,3)=1$	$J_A(1,4)=0$ $J_B^{*}(1,4)=20$
$u_{\text{Player 1}}=2$	$J_A(2,1)=1$ $J_B(2,1)=7$	$J_A^{*}(2,2)=10$ $J_B^{*}(2,2)=8$	$J_A(2,3)=4$ $J_B(2,3)=4$	$J_A(2,4)=6$ $J_B(2,4)=2$
$u_{\text{Player 1}}=3$	$J_A(3,1)=4$ $J_B(3,1)=6$	$J_A(3,2)=2$ $J_B^{*}(3,2)=7$	$J_A^{*}(3,3)=20$ $J_B(3,3)=3$	$J_A^{*}(3,4)=8$ $J_B(3,4)=3$
$u_{\text{Player 1}}=4$	$J_A(4,1)=3$ $J_B^{*}(4,1)=10$	$J_A(4,2)=9$ $J_B(4,2)=2$	$J_A(4,3)=16$ $J_B(4,3)=5$	$J_A(4,4)=7$ $J_B(4,4)=5$

Assume that the action - reaction search begins by setting Player 2's initial strategy such that $u^0_{\text{Player 2}} = 1$. Player 1 then calculates its optimal response to this strategy from Player 2. In the game shown in Table 4.1, Player 1's optimal response to Player 2's strategy $u^0_{\text{Player 2}} = 1$ is strategy #1 making $u^1_{\text{Player 1}} = 1$. Then Player 2's optimal response to Player 1's strategy $u^1_{\text{Player 1}} = 1$ is found to be strategy # 4 making $u^1_{\text{Player 2}} = 4$. This process then proceeds as follows:

$$\begin{array}{cc} & u^0_{\text{Player 2}} = 1 \\ u^1_{\text{Player 1}} = 1 & u^1_{\text{Player 2}} = 4 \\ u^2_{\text{Player 1}} = 3 & u^2_{\text{Player 2}} = 2 \\ u^3_{\text{Player 1}} = 2 & u^3_{\text{Player 2}} = 2 \end{array} \tag{4.6}$$

Here we see that because Player 2 repeated strategies in consecutive steps, the action - reaction search has converged according to the first criteria, that of the Nash equilibrium. This process can be further shown on Table 4.2.

Table 4.2 – Example of the Strategies considered in the Action - reaction Search

	$u_{\text{Player 2}} = 1$	$u_{\text{Player 2}} = 2$	$u_{\text{Player 2}} = 3$	$u_{\text{Player 2}} = 4$
$u_{\text{Player 1}} = 1$	$J_A^*(1,1) = 5$ $J_B(1,1) = 15$	$J_A(1,2) = 6$ $J_B(1,2) = 4$	$J_A(1,3) = 14$ $J_B(1,3) = 1$	$J_A(1,4) = 0$ $J_B^*(1,4) = 20$
$u_{\text{Player 1}} = 2$	$J_A(2,1) = 1$ $J_B(2,1) = 7$	$J_A^*(2,2) = 10$ $J_B^*(2,2) = 8$	$J_A(2,3) = 4$ $J_B(2,3) = 4$	$J_A(2,4) = 6$ $J_B(2,4) = 2$
$u_{\text{Player 1}} = 3$	$J_A(3,1) = 4$ $J_B(3,1) = 6$	$J_A(3,2) = 2$ $J_B^*(3,2) = 7$	$J_A^*(3,3) = 20$ $J_B(3,3) = 3$	$J_A^*(3,4) = 8$ $J_B(3,4) = 3$
$u_{\text{Player 1}} = 4$	$J_A(4,1) = 3$ $J_B^*(4,1) = 10$	$J_A(4,2) = 9$ $J_B(4,2) = 2$	$J_A(4,3) = 16$ $J_B(4,3) = 5$	$J_A(4,4) = 7$ $J_B(4,4) = 5$

In the action - reaction search we may use any strategy as the initial strategy[4]. For example if we assume that $u^0_{\text{Player 2}} = 2$ we converge at the Nash equilibrium at the end of the first iteration as $u^1_{\text{Player 1}} = 2$ and $u^2_{\text{Player 1}} = 2$. The action - reaction search will also converge at the Nash equilibrium at the end of the 2nd iteration when $u^0_{\text{Player 2}} = 3$ and also when $u^0_{\text{Player 2}} = 4$. The action - reaction search also works when an initial strategy is chosen for Player 1. $u^0_{\text{Player 2}} = 4$. In this example, no matter the initial strategy, the action - reaction search will always converge on the Nash equilibrium.

In the previous example, there is little difference between the required complexity of the exhaustive search and the action - reaction search. This is because of the small scale of the problem. To illustrate the dramatic improvement in speed recall the example given in Figure 2.2 reprinted in Figure 4.1. Here we can see that the action - reaction search converges upon one of the two Nash equilibrium points before the end of the second iteration for any initial strategy on the Red Team and before the end of the third iteration for any initial strategy of the Blue Team. Clearly, this represents a tremendous improvement in the number of computations required to find a pair of Nash strategies.

[4] This only applies to the player setting the initial strategy. Because the second player will select its optimal response to the initial strategy of the first player in the action - reaction search, there is no need to ever assign the second player an initial strategy.

Blue Weapon Target Assignment Strategies	Red Weapon Target Assignment Strategies																
		{1,1;1,1}	{1,2;1,1}	{2,1;1,1}	{2,2;1,1}	{1,1;1,2}	{1,2;1,2}	{2,1;1,2}	{2,2;1,2}	{1,1;2,1}	{1,2;2,1}	{2,1;2,1}	{2,2;2,1}	{1,1;2,2}	{1,2;2,2}	{2,1;2,2}	{2,2;2,2}
	{1,1;1,1}	0.10	0.26	0.58	0.45	0.24	0.37	0.47	0.29	0.33	0.44	0.47	0.29	0.42	0.52	0.31	0.05
		-0.10	-0.26	-0.58	-0.45	-0.24	-0.37	-0.47	-0.29	-0.33	-0.44	-0.47	-0.29	-0.42	-0.52	-0.31	-0.05
	{1,2;1,1}	-0.54	-0.36	0.05	-0.08	-0.51	-0.34	0.04	-0.10	-0.10	-0.01	-0.24	-0.51	-0.09	0.00	-0.27	-0.56
		0.54	0.36	-0.05	0.08	0.51	0.34	-0.04	0.10	0.10	0.01	0.24	0.51	0.09	0.00	0.27	0.56
	{2,1;1,1}	-0.55	-0.38	-0.02	-0.16	-0.51	-0.35	-0.04	-0.19	-0.18	-0.08	-0.26	-0.52	-0.16	-0.06	-0.30	-0.57
		0.55	0.38	0.02	0.16	0.51	0.35	0.04	0.19	0.18	0.08	0.26	0.52	0.16	0.06	0.30	0.57
	{2,2;1,1}	-0.47	-0.26	0.41	0.33	-0.47	-0.25	0.41	0.33	0.26	0.33	-0.10	-0.42	0.26	0.33	-0.11	-0.43
		0.47	0.26	-0.41	-0.33	0.47	0.25	-0.41	-0.33	-0.26	-0.33	0.10	0.42	-0.26	-0.33	0.11	0.43
	{1,1;1,2}	-0.55	-0.30	0.44	0.33	-0.41	-0.19	0.33	0.17	-0.33	-0.11	0.34	0.16	-0.23	-0.04	0.17	-0.07
		0.55	0.30	-0.44	-0.33	0.41	0.19	-0.33	-0.17	0.33	0.11	-0.34	-0.16	0.23	0.04	-0.17	0.07
	{1,2;1,2}	-0.41	-0.25	0.06	-0.07	-0.38	-0.23	0.05	-0.09	0.03	0.10	-0.23	-0.51	0.04	0.11	-0.26	-0.55
		0.41	0.25	-0.06	0.07	0.38	0.23	-0.05	0.09	-0.03	-0.10	0.23	0.51	-0.04	-0.11	0.26	0.55
	{2,1;1,2}	-0.50	-0.34	-0.03	-0.17	-0.47	-0.31	-0.05	-0.20	-0.14	-0.04	-0.26	-0.52	-0.12	-0.03	-0.31	-0.58
		0.50	0.34	0.03	0.17	0.47	0.31	0.05	0.20	0.14	0.04	0.26	0.52	0.12	0.03	0.31	0.58
	{2,2;1,2}	-0.03	0.12	0.47	0.37	-0.02	0.12	0.47	0.37	0.71	0.71	-0.04	-0.38	0.71	0.71	-0.05	-0.39
		0.03	-0.12	-0.47	-0.37	0.02	-0.12	-0.47	-0.37	-0.71	-0.71	0.04	0.38	-0.71	-0.71	0.05	0.39
	{1,1;2,1}	0.05	0.19	0.10	-0.19	0.19	0.30	-0.01	-0.35	0.28	0.38	-0.01	-0.35	0.37	0.45	-0.17	-0.59
		-0.05	-0.19	-0.10	0.19	-0.19	-0.30	0.01	0.35	-0.28	-0.38	0.01	0.35	-0.37	-0.45	0.17	0.59
	{1,2;2,1}	-0.54	-0.36	0.10	-0.01	-0.52	-0.34	0.09	-0.03	-0.10	-0.01	-0.19	-0.44	-0.09	0.00	-0.22	-0.49
		0.54	0.36	-0.10	0.01	0.52	0.34	-0.09	0.03	0.10	0.01	0.19	0.44	0.09	0.00	0.22	0.49
	{2,1;2,1}	-0.55	-0.38	-0.03	-0.17	-0.52	-0.35	-0.05	-0.19	-0.18	-0.09	-0.26	-0.52	-0.17	-0.07	-0.31	-0.58
		0.55	0.38	0.03	0.17	0.52	0.35	0.05	0.19	0.18	0.09	0.26	0.52	0.17	0.07	0.31	0.58
	{2,2;2,1}	-0.46	-0.24	0.65	0.66	-0.45	-0.23	0.65	0.66	0.27	0.35	0.14	-0.09	0.27	0.35	0.13	-0.11
		0.46	0.24	-0.65	-0.66	0.45	0.23	-0.65	-0.66	-0.27	-0.35	-0.14	0.09	-0.27	-0.35	-0.13	0.11
	{1,1;2,2}	-0.55	-0.31	0.06	-0.20	-0.42	-0.20	-0.05	-0.36	-0.33	-0.13	-0.05	-0.36	-0.23	-0.05	-0.21	-0.59
		0.55	0.31	-0.06	0.20	0.42	0.20	0.05	0.36	0.33	0.13	0.05	0.36	0.23	0.05	0.21	0.59
	{1,2;2,2}	-0.40	-0.23	0.15	0.04	-0.37	-0.21	0.13	0.02	0.05	0.12	-0.14	-0.40	0.06	0.13	-0.18	-0.44
		0.40	0.23	-0.15	-0.04	0.37	0.21	-0.13	-0.02	-0.05	-0.12	0.14	0.40	-0.06	-0.13	0.18	0.44
	{2,1;2,2}	-0.49	-0.33	0.01	-0.13	-0.46	-0.30	-0.01	-0.16	-0.12	-0.03	-0.23	-0.49	-0.11	-0.02	-0.27	-0.54
		0.49	0.33	-0.01	0.13	0.46	0.30	0.01	0.16	0.12	0.03	0.23	0.49	0.11	0.02	0.27	0.54
	{2,2;2,2}	0.00	0.15	0.74	0.74	0.01	0.16	0.74	0.74	0.74	0.75	0.24	-0.01	0.74	0.75	0.23	-0.02
		0.00	-0.15	-0.74	-0.74	-0.01	-0.16	-0.74	-0.74	-0.74	-0.75	-0.24	0.01	-0.74	-0.75	-0.23	0.02

Figure 4.1 – Example of a Game Matrix for the 2x2 MT-DWTA

In all previous examples, we considered scenarios that had Nash equilibrium points. As we mentioned earlier, the MT-DWTA is not guaranteed to have any Nash strategies. Consider the game matrix presented in Table 2.4 and reprinted in Table 4.3.

Table 4.3 – Example of a Game Matrix Without a Set of Nash Strategies

	$u_{\text{Player 2}}=1$	$u_{\text{Player 2}}=2$	$u_{\text{Player 2}}=3$	$u_{\text{Player 2}}=4$
$u_{\text{Player 1}}=1$	$J_1^*(1,1)=5$ $J_2(1,1)=15$	$J_1(1,2)=6$ $J_2(1,2)=4$	$J_1(1,3)=14$ $J_2(1,3)=1$	$J_1(1,4)=0$ $J_2^*(1,4)=20$
$u_{\text{Player 1}}=2$	$J_1(2,1)=1$ $J_2(2,1)=7$	$J_1(2,2)=8$ $J_2^*(2,2)=8$	$J_1(2,3)=4$ $J_2(2,3)=4$	$J_1(2,4)=6$ $J_2(2,4)=2$
$u_{\text{Player 1}}=3$	$J_1(3,1)=4$ $J_2(3,1)=6$	$J_1(3,2)=2$ $J_2^*(3,2)=7$	$J_1^*(3,3)=20$ $J_2(3,3)=3$	$J_1^*(3,4)=8$ $J_2(3,4)=3$
$u_{\text{Player 1}}=4$	$J_1(4,1)=3$ $J_2^*(4,1)=10$	$J_1^*(4,2)=9$ $J_2(4,2)=2$	$J_1(4,3)=16$ $J_2(4,3)=5$	$J_1(4,4)=7$ $J_2(4,4)=5$

If the initial strategy for Player 2 is again assumed to be $u^0_{\text{Player 2}}=1$ the following progression can be observed:

$$\begin{array}{ll} & u^0_{\text{Player 2}}=1 \\ u^1_{\text{Player 1}}=1 & u^1_{\text{Player 2}}=4 \\ u^2_{\text{Player 1}}=3 & u^2_{\text{Player 2}}=2 \\ u^3_{\text{Player 1}}=4 & u^3_{\text{Player 2}}=1 \end{array} \tag{4.7}$$

Here the action - reaction algorithm terminated because Player 2 repeated strategy #1 in the 0^{th} and the 3^{rd} iteration, satisfying the second convergence criteria. It is difficult to justify any strategy in this case because a Nash strategy was not found. The concept of a Nash equilibrium needs to be extended before the selection of a strategy can be justified.

4.2 A GENERALIZATION OF THE NASH EQUILIBRIUM

Often, game theoretic scenarios do not have a precise Nash equilibrium. Such problems would benefit from a more general concept of Nash equilibrium in the same manner that optimization problems benefit from the concept of optimality when an optimal solution cannot be found. The largest impediment to such a concept is that Nash strategies are not optimal in the same sense of a standard optimization problem. As mentioned earlier, strategies near the optimal strategy can be assumed to be near optimal in most cases, a property not shared by Nash equilibrium. Many optimization problems can often be assumed to be convex in the area around the optimum while the Nash equilibrium is only defined for strategies that satisfy the Nash equilibrium criteria. No concept of "closeness" to the Nash equilibrium exists in the typical Nash definition. To solve this problem and generate a concept of closeness to the Nash Equilibrium we formulate the Nash equilibrium as a form of a general optimization problem.

Recall the definition of the Nash equilibrium conditions given in section 2.1. A pair of strategies $u^N_{\text{Player 1}}, u^N_{\text{Player 2}}$ are Nash strategies if they satisfy the following two conditions:

$$\begin{aligned} J_{\text{Player 1}}\left(u^N_{\text{Player 1}}, u^N_{\text{Player 2}}\right) &\geq J_A\left(u_{\text{Player 1}}, u^N_{\text{Player 2}}\right) \quad \text{for all } u_{\text{Player 1}} \in U_{\text{Player 1}} \\ J_{\text{Player 2}}\left(u^N_{\text{Player 1}}, u^N_{\text{Player 2}}\right) &\geq J_B\left(u^N_{\text{Player 1}}, u_{\text{Player 2}}\right) \quad \text{for all } u_{\text{Player 2}} \in U_{\text{Player 2}} \end{aligned} \tag{4.8}$$

Writing Player 1's optimal response to a given strategy of Player 2, say $u_{\text{Player 2}}$, as:

$$J_{\text{Player 1}}\left(u^*_{\text{Player 1}}\left(u_{\text{Player 2}}\right), u_{\text{Player 2}}\right) = \max_{U_{\text{Player 1}}} J_{\text{Player 1}}\left(u_{\text{Player 1}}, u_{\text{Player 2}}\right) \tag{4.9a}$$

and likewise writing Player 2's optimal response to a given strategy of Player 1, say $u_{\text{Player 1}}$, as:

$$J_{\text{Player 2}}\left(u_{\text{Player 1}}, u^{*}_{\text{Player 2}}\left(u_{\text{Player 1}}\right)\right) = \max_{U_{\text{Player 2}}} J_{\text{Player 2}}\left(u_{\text{Player 1}}, u_{\text{Player 2}}\right) \tag{4.9b}$$

yields the following when combined with (4.8):

$$\begin{aligned} J_{\text{Player 1}}\left(u^{*}_{\text{Player 1}}\left(u^{N}_{\text{Player 2}}\right), u^{N}_{\text{Player 2}}\right) - J_{\text{Player 1}}\left(u^{N}_{\text{Player 1}}, u^{N}_{\text{Player 2}}\right) = 0 \\ J_{\text{Player 2}}\left(u^{N}_{\text{Player 1}}, u^{*}_{\text{Player 2}}\left(u^{N}_{\text{Player 1}}\right)\right) - J_{\text{Player 2}}\left(u^{N}_{\text{Player 1}}, u^{N}_{\text{Player 2}}\right) = 0 \end{aligned} . \tag{4.10}$$

The expressions in (4.10) simply state that no player can improve their objective score over that obtained at a Nash equilibrium assuming that their opponent also remains at a Nash equilibrium, the explicit definition of the Nash equilibrium. The expressions in (4.10) also suggest that a Nash-like strategy is one in which neither player can improve their objective significantly from a given set of strategies. Extending the framework of (4.10), we can formulate the Nash equilibrium as an optimization problem, minimizing the difference between a pair of strategies and their corresponding optimal responses. For example consider the following minimization:

$$\begin{aligned} \min_{u_{\text{Player 1}}, u_{\text{Player 2}}} \Big[J_{\text{Player 1}}\left(u^{*}_{\text{Player 1}}\left(u_{\text{Player 2}}\right), u_{\text{Player 2}}\right) - J_{\text{Player 1}}\left(u_{\text{Player 1}}, u_{\text{Player 2}}\right) \Big]^2 + \\ \Big[J_{\text{Player 2}}\left(u_{\text{Player 1}}, u^{*}_{\text{Player 2}}\left(u_{\text{Player 1}}\right)\right) - J_{\text{Player 2}}\left(u_{\text{Player 1}}, u_{\text{Player 2}}\right) \Big]^2 \end{aligned} \tag{4.11}$$

Here we see that when $u_{\text{Player 1}} = u^{N}_{\text{Player 1}}$ and $u_{\text{Player 2}} = u^{N}_{\text{Player 2}}$, the case of a Nash equilibrium, the expression given in (4.11) reduces to 0, the lowest value possible. That is to say that a pair of strategies is a Nash pair of strategies if neither player has an incentive to unilaterally alter their strategy. The expression given in (4.11) also provides a basis for a game theoretic solution in scenarios which do not have a Nash equilibrium. By finding a pair of strategies which minimize (4.11), we find the pair of strategies which most closely resemble a Nash equilibrium.

While it is difficult to justify strategies found by minimizing (4.11) because such a process inherently relies on the assumption that one's opponent is selecting a strategy in a similar manner, the same objections can be made of the Nash equilibrium. There is no way to predict

the optimality of the outcome in a non-zero sum game if one player selects a Nash strategy and the other player selects a different strategy. It is clear that the player who fails to select a Nash strategy will be penalized, but fate of that player's opponent is uncertain. The same holds true for strategies that are similar to Nash strategies only with greater uncertainty as to the outcome.

We make use of this definition of the Nash equilibrium given in (4.11) to solve for cases in the action - reaction search such as (4.5) where the search converges to a series of individual strategies. Using this measure, we employ the most "Nash-like" of the given strategies. The concept of closeness is also useful for applying the action - reaction search to game scenarios with more than two players. Instead of alternating, Player 1, Player 2, Player 3, etc…., we can make use of the concept of closeness. At every iteration we allow each team to vary its strategy using ULTRA, implementing the change which minimizes (4.11). In that way, we obtain a set of strategies that is closer to a Nash equilibrium at ever iteration.

4.3 JUSTIFYING THE NASH EQULIBRIUM IN A MT-DWTA GAME

The Nash strategy in game theory has often been criticized as being ineffective in competitive multi-team target assignment problems, especially when compared to the random or greedy targeting strategies. Using a set of experiments, we will show that this is not the case. We consider an attrition model of two teams of non-homogeneous fighting units simultaneously targeting each other and we compare the outcomes when various combinations of four targeting strategies are employed by each team. The four strategies are: (1) The Random strategy where each unit selects its target randomly, (2) The Unit Greedy strategy where each unit chooses the

target that optimizes its own performance only, (3) The Team Optimal strategy where the units coordinate their choice of targets so as to optimize the overall team performance without considering the choice of strategies by the other team, and (4) the Team Nash strategy where the units coordinate their choice of targets so as to optimize the overall team performance but now taking into consideration the choice of strategies by the other team. We will then compare the results for all 16 possible combinations of these targeting strategies and show that for each team the Nash strategy outperforms all other strategies no matter what is the strategy employed by the other side.

The multi-team target assignment problem considers two or more teams of fighting units. For simplicity, in the following experiments we will consider only the case of two teams. To present a mathematical formulation for this structure, let the two teams be labeled as Blue and Red and let K denote the total number of time steps representing the duration of the battle. Let the number of non-homogeneous fighting units at step k, where $k = 0,1,....,K-1$, in each team be $N_B(k)$ and $N_R(k)$ respectively. Recall the definition of the MT-DWTA given in (1.7). If, at each battle step k, a team chooses a strategy based upon an objective function, we assume that this objective function will take the form of a weighted sum, maximizing both the combined worth of the destroyed units in the other team and the combined worth of the remaining units in that team. Let these objective functions at step k be $J_B\left(u_B(k), u_R(k), k\right)$ for the Blue team and $J_R\left(u_B(k), u_R(k), k\right)$ for the Red team. In our model, each unit may be valued differently by each team. Let b_i^B denote the worth of the i^{th} Blue unit to the Blue team and let b_i^R denote the worth of that unit to the Red team. Likewise, let r_j^B and r_j^R denote the worth of the j^{th} Red unit to the Blue and Red teams respectively. Assume that the probability of kill of the i^{th} Blue unit against

the j^{th} Red unit is $p^B_{i,j}$. Similarly, let the probability of kill of the j^{th} Red unit against the i^{th} Blue unit be $p^R_{j,i}$. Finally, let $B_i(k)$ and $R_j(k)$ denote the probabilities that the i^{th} Blue unit and j^{th} Red unit are alive at the start of the k^{th} battle step. Using these notations we can express the probability of survival of the i[th] Blue unit and j[th] Red unit as follows:

$$B_i(k) = B_i(k-1)\prod_{j=1}^{N_R(k)}\left[1 - p^R_{j,i}\delta\left(i - u_{Rj}(k)\right)R_j(k-1)\right] \tag{4.12a}$$

$$R_j(k) = R_j(k-1)\prod_{i=1}^{N_B(k)}\left[1 - p^B_{i,j}\delta\left(u_{Bi}(k) - j\right)B_i(k-1)\right] \tag{4.12b}$$

The objective functions for the Blue and Red teams are expressed as follows:

$$J_B(u_B, u_R, k) = \sum_{i=1}^{N_B(k)} b_i^B B_i(k) - \sum_{j=1}^{N_R(k)} r_j^B R_j(k) \tag{4.13a}$$

$$J_R(u_B, u_R, k) = -\sum_{i=1}^{N_B(k)} b_i^R B_i(k) + \sum_{j=1}^{N_R(k)} r_j^R R_j(k) \tag{4.14b}$$

Recall that the Random Target Assignment strategy and the Unit Greedy Target Assignment strategy were defined in Chapter 3. We now define the remaining two strategies mentioned earlier:

c) Definition 4: A strategy $u_B^o(k) \in U_B^o(k)$ is called a **Team Optimal strategy** for the Blue team at step k if it is selected such that $J_B(u_B^o(k), \underset{\sim}{0}, k) \geq J_B(u_B(k), \underset{\sim}{0}, k)$ for all $u_B(k) \in U_B^o(k)$. Similarly, a vector $u_R^o(k) \in U_R^o(k)$ is called a **Team Optimal strategy** for the Red team if it is selected such that $J_R(\underset{\sim}{0}, u_R^o(k), k) \geq J_R(\underset{\sim}{0}, u_R(k), k)$ for all $u_R(k) \in U_R^o(k)$.

We note that the Team Optimal strategy is one that completely ignores the adversarial nature of the other team and considers it only as a set of target units. It represents the standard non-game based solution to the target assignment problem.

The definition of the Team Nash strategies, however, is more complex. This strategy requires that each team must make the assumption that its adversary is also using a Nash strategy, whether or not that its adversary actually uses this strategy. Another consideration is that at the current battle step *k*, the two objective functions in (2) are decoupled in that each team's objective function does not depend on the other team's strategy. Hence, their maximization at the current battle step *k* only will yield the Team Optimal strategies as defined earlier. Consequently, a game in this problem can only be defined if the objective functions are projected over a horizon of two or more battle steps. That is, at each step *k* the teams will determine their strategies not only for step *k* but also for steps $k+1,\ k+2,\ldots\ldots,\ k+d-1$ where *d* represents the depth of the look-ahead horizon. As we mentioned earlier, a game will exist only if $d \geq 2$. In this section, for the sake of simplicity, we will consider only Team Nash strategies defined over a look-ahead horizon $d = 2$.

d) Definition 5: A pair of strategies $\{u_B^N(k),\ u_B^N(k+1)\} \in U_B^N(k)$ and $\{u_R^N(k),\ u_R^N(k+1)\} \in U_R^N(k)$ defined over a look-ahead horizon $d = 2$ is called a **Team Nash** pair of strategies at step *k* if it satisfies the inequalities:

$$\begin{aligned} &J_B\left(u_B^N(k+1), u_R^N(k+1), k+1\right) \geq J_B\left(u_B(k+1), u_R^N(k+1), k+1\right) \\ &\qquad \forall\ \{u_B(k),\ u_B(k+1)\} \in U_B^N(k) \\ &J_R\left(u_B^N(k+1), u_R^N(k+1), k+1\right) \geq J_R\left(u_B^N(k+1), u_R(k+1), k+1\right) \\ &\qquad \forall\ \{u_R(k),\ u_R(k+1)\} \in U_R^N(k) \end{aligned} \tag{4.15}$$

It is important at this stage to make the following two remarks:

1. As is clear from (4.15), at each stage k the teams' objectives are to maximize the objective functions (4.14) at step k+1, which is the end of the look-ahead horizon. The explicit dependence of these functions on both strategies at *k* and *k*+1 is evident from expression (4.13).

2. Note that even though the Nash strategies are computed for the two consecutive steps *k* and *k*+1, only the strategies at step *k* are implemented and the process repeated at step *k*+1.

In order to compare the effectiveness of the four different target selection strategies mentioned earlier, the Random, Unit Greedy, Team Optimal and Team Nash Target Assignment strategies, we consider the following scenario. At $k=0$, the Blue team consists of $N_B^0=10$ units and the Red team consists of $N_R^0=10$. We will perform two experiments. In the first we leave the problem unstructured, assuming the probabilities of kill and the unit worth values to be uniform random numbers. In the second, we consider a more structured (i.e. more realistic) scenario and assign specific probabilities of kill and units worth values to emphasize different roles for different units in each team. In both experiments, we use a Monte Carlo approach to determine battle damage. For example, if a Blue unit *i* was assigned a Red targets *j* and the corresponding probability of kill is $p_{i,j}^B=0.7$, then our simulator selects a random number in the interval [0, 1]. If the number is greater than 0.7, the Red unit survives, if not the Red unit is destroyed. This process was implemented at every battle step, stopping only when all units in one team are completely destroyed or after 6 successive battle steps. Because of the dimensionalities of the search spaces for the Team Optimal and Team Nash strategies ($S_B^o=11^{10}$, $S_R^o=11^{10}$,and

$S_B^N = S_R^N = 14.641 \times 10^{43}$ respectively), we used the ULTRA algorithm with $F = 1$ to determine these strategies. For example: an exhaustive search would have required 10^{35} times the computer time required by ULTRA. Using a 2.2 GHz Pentium processor, the ULTRA algorithm took about 12 hours of computer time to determine the Nash strategies. An exhaustive search would have required 4.5×10^{27} years!

4.3.1 Experiment IV

In this experiment, our objective is to get completely unbiased results. We assume that the two 10×10 matrices of probabilities of kill of Blue against Red and Red against Blue have entries that are random numbers uniformly distributed in the interval [0, 1]. The objective functions for each team were structured as described in equations (4.13) and (4.14) and the unit worth values $b_i^B, b_i^R, r_j^B, r_j^R$ were also randomly and independently selected in the range [0, 1] with uniform probability distributions. To obtain valid aggregate results, we performed 30,000 runs for each of the 16 possible combinations of strategies and averaged the results. These runs differed in that all randomly generated parameters were selected for each run using a different seed. The results of this experiment are tabulated in Table 4.4. This table shows the average percentage of the initial force remaining at the end of the battle for each of the 16 combinations of strategies. As expected, the outcomes are balanced when both teams use the same strategy. The largest remaining force occurs if both sides use Unit Greedy strategies. The smallest remaining force occurs when both sides use Team Nash strategies. However, when faced with not knowing the opponent's strategy, the Unit Greedy strategy appears to be the worst and the Team Nash

strategy appears to be the smartest. The Random and Team Optimal strategies are somewhere in between. For example, if unknown to Blue, the Red uses a Random strategy then, among all the choices that Blue has, the Team Nash strategy appears to be the best. These results indicate that the Team Nash strategy yields on average a superior outcome independent of the adversary's strategy. An interesting observation that can also be observed from Table 4.4 is that while the Team Nash is the best strategy in all cases, the Team Optimal strategy appears to provide almost as good results. This is due to the total randomness of this scenario and the advantage of the Team Nash strategy should be more significant in a structured scenario as will be illustrated in the next experiment.

Table 4.4 - Comparison of average percentage of initial force remaining per team (Blue over Red) at the end of battle for different strategies and for the scenario of Experiment IV

		Red Strategy			
		Unit Random	Unit Greedy	Team Optimal	Team Nash
Blue Strategy	Unit Random	15%	33%	0%	0%
		15%	3%	47%	50%
	Unit Greedy	3%	17%	0%	0%
		33%	18%	53%	54%
	Team Optimal	48%	53%	9%	7%
		0%	0%	9%	12%
	Team Nash	49%	54%	12%	9%
		0%	0%	7%	9%

4.3.2 Experiment V

In this experiment, we add considerable structure to the scenario. We assume that the Blue team now consists of 3 Light, 3 Medium and 3 Heavy fighting units and that the Red team consists of

5 High Value (HV) units and 5 Defender units. The probabilities of kill and unit worth now have specific values as indicated in Tables 4.5 and 4.6. Because of our use of a Monte Carlo approach to determine battle damage, we again performed 30,000 runs for each of the 16 possible combinations and averaged the results. The average percentages of the initial force remaining at the end of the battle for each of the 16 combinations of strategies are shown in Table 4.7.

Table 4.5 - Blue Team probabilities of kill against Red team and unit worth to both Blue and Red.

BLUE TEAM				
	Probability of Kill		Unit Worth	
Type of Unit	Against Red HV units	Against Red Defenders	To Blue	To Red
Light: Units 1, 2, and3	0.5	0.4	5	9
Medium: Units 4, 5, and 6	0.4	0.5	10	10
Heavy: Units 7, 8, 9, and 10	0.5	0.6	15	11

Table 4.6 – Red Team Probabilities of Kill against Blue Team and Unit Worth to both Blue and Red.

RED TEAM					
	Probability of Kill			Unit Worth	
Type of Unit	Against Blue Light	Against Blue Medium	Against Blue Heavy	To Red	To Blue
High Value: Units 1, 2, 3, 4, and 5)	0.1	0.05	0	15	14
Defenders: Units 6, 7, 8, 9, and 10	0.9	0.85	0.75	7	6

First, we notice that the Unit Greedy strategy fares even worse in this more structured scenario than in the scenario of Experiment IV. A team using the Unit Greedy strategy will be completely

destroyed unless its opponent also uses the Unit Greedy strategy. This is so because it is highly likely that many units will share the same optimal target, resulting in many units attacking a single target while ignoring the rest. Second, unlike the previous experiment, here we see that there is a significant gain for the Blue Team when it uses a Team Nash strategy as compared to a Team Optimal strategy. In contrast, the Red Team gains very little by implementing a Team Nash strategy as compared to a Team Optimal strategy. This illustrates that the Team Nash strategy is at least as good as the Team Optimal strategy and can provide a significant improvement depending on the scenario.

Table 4.7 - Comparison of average percentage of initial force remaining per team (Blue over Red) at the end of battle for different strategies and for the scenario of Experiment V.

		Red Strategy			
		Unit Random	Unit Greedy	Team Optimal	Team Nash
Blue Strategy	Unit Random	24%	71%	12%	11%
		9%	0%	19%	21%
	Unit Greedy	0%	36%	0%	0%
		65%	41%	73%	74%
	Team Optimal	40%	78%	24%	27%
		4%	0%	8%	10%
	Team Nash	48%	80%	34%	35%
		3%	0%	9%	10%

4.3.3 Experiment VI

In the previous two experiments, we have established that our implementation of the Nash equilibrium, using the action - reaction search combined with ULTRA has resulted in a

statistically significant improvement over standard optimization techniques. Due to the limitations of the exhaustive search however, we can not directly calculate or confirm that the strategies we employed were actually Nash equilibrium points in the previous. It is inherently difficult if not impossible to even confirm that such equilibrium actually exists. To rectify this, in Experiment VI we sought to ascertain how closely the strategies we employed compared to Nash strategies.

To evaluate how favorably the strategies we employed compared to Nash equilibrium points, we used a similar scenario to that in Experiment IV in that we again confined ourselves to two heterogeneous teams. We assumed that the two matrices of probabilities of kill of Blue against Red and Red against Blue have entries that are random numbers uniformly distributed in the interval [0, 1] and that the unit worth values $b_i^B, b_i^R, r_j^B, r_j^R$ were also randomly and independently selected in the range [0, 1], each with uniform probability distributions. To obtain valid aggregate results, we performed 30,000 runs for each of the scenario and averaged the results. These runs differed in that all random numbers were selected for each run using a different seed.

While Experiment IV was designed to compare the results when the two teams employ combinations of four different types of strategies, Experiment VI was designed to measure how closely the Nash strategies that we employed corresponded to the Nash equilibrium. To measure this, we first calculated the Nash strategies for both teams and compiled the results. We then calculated the Red team's optimal reaction to a Nash strategy for the Blue Team. By comparing these results with those obtained when both teams employed a Nash strategy it is possible to determine how closely our implementation to calculate a Nash strategy corresponds to the

definition of the Nash equilibrium. Table 4.8 compiles these results for the case when there are 10 units on both the Red and Blue teams.

Table 4.8 – Measuring the closeness to a Nash strategy, 10 vs 10 when Blue uses a Nash

		Strategies Used		
		Nash vs Nash (Red)	Nash vs Optimal Reaction (Red)	Nash vs Team Optimal (Red)
% Remain	Blue	9%	8%	12%
	Red	9%	10%	7%

Table 4.8 illustrates that our implementation achieves results that are very near to that of a true Nash implementation. When the Red Team is unfairly allowed to know the Blue Team's strategy in advance, it manages to only destroy 1% more Blue Units and preserve 1% more of its own. In contrast, when Blue employs the best Naïve strategy, the Team Optimal Strategy, Red destroys 2% more Blue units and preserves 3% more of its own. This is especially notable because it is strongly suspected that many instances of the above experiment did not actually have a unique Nash solution. Consequently, it is likely that our implementation often arrives at a Nash strategy when such equilibrium exists, and generates a result similar to Nash equilibrium, as defined earlier, when such a Nash equilibrium does not exist.

5.0 USING THE MT-DWTA MODEL TO CREATE A TDT CONTROLLER

Modern military conflict combines a near infinite number of strategies and command decisions, considerable heterogeneity and interdependency in unit attributes and pervasive uncertainty in regards to units on both sides. The resulting complexity of military engagements creates problems far outside the human capacity to comprehend and process completely. To cope with human limitations, military planners throughout the ages have broken battle management into hierarchical stages, with commanders planning the battle at varying degrees of abstraction. By thus reducing the complexity of the command decisions, individual commanders are able to make intelligent decisions based upon the types of information necessary for their particular rung in the hierarchical ladder while ignoring mostly irrelevant data. However, while the simplification of battle data does aid in the planning process, individual commanders are still required to process massive amounts of data and make decisions in highly uncertain environments. Mistakes often occur when commanders are overloaded with data, stress and lack of sleep. One promising method to reduce the number of mistakes and to increase the efficiency of commanders is to use mixed initiative automated controllers [27-30]. These computerized planners aid commanders by suggesting various possible strategies, estimating the outcome of a when a particular strategy is employed and directly controlling portions of the battle as the commander sees fit. As an added benefit, because these controllers are often mixed initiative in nature, commanders can control the battle at any level of coarseness they deem appropriate, from

controlling individual units to setting global objectives. Such advantages are further stressed by the impending immergence of unmanned vehicles, with automated planners reduce the support network required to manage large numbers of unmanned combat air vehicles (UCAVs).

One recent attempt to produce a viable mixed initiative automated planner was undertaken by DARPA under the Mixed Initiative Control for Automa-teams (MICA) project [31]. The goal of this project was to create a robust automated battle planner with variable human initiative control designed to operate on the Boeing OEP, a realistic combat computer simulation. This simulation package included full 3D coordinates including terrain effects, sensor data, weapon effectiveness and damage assessment.

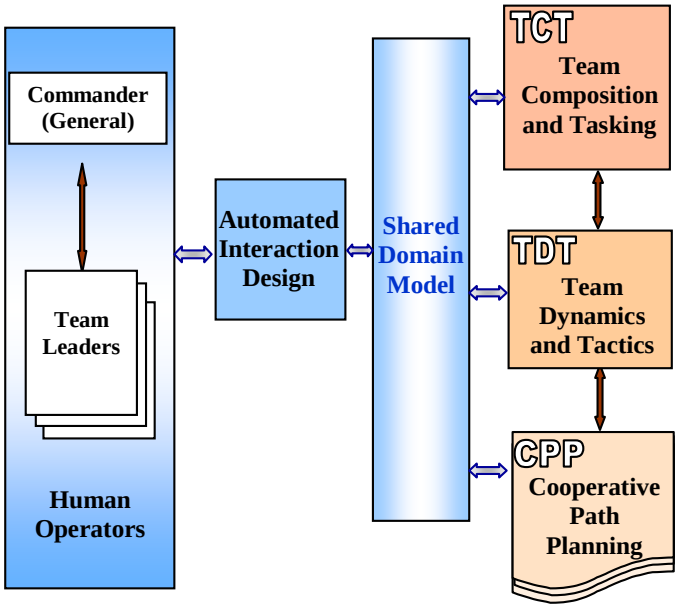

Figure 5.1 – Hierarchical Architecture of SHARED

One attempt to make such a controller was the Strategies for Human-Automaton Resource Entity Development (SHARED) project[32-35]. This project is significant because it used game theoretic solution concepts throughout the entirety of its control decisions. Modeled after actual

battlefield control procedures, the SHARED control scheme is hierarchical. The overall architecture of the SHARED project is illustrated in Figure 5.1.

The SHARED hierarchy operates by efficiently distributing information and command decisions where they are best handled. Facilitating this is the SHARED Domain Model. This is a database designed to retain and output all relevant information available from the OEP and employ control decisions made by the human and autonomous controllers. The SHARED domain model also serves another function as it allows for easy interfacing with other battlefield models. Instead of a complete software rewrite, it is only necessary to write an interface between any given battlefield database and the SHARED domain model. Connected to the SHARED Domain Model is the Automated Interaction Design (AID) package[36] which examines the available data and the necessary tasks and presents human users with a customized user interface, allowing users to access the data and controls most appropriate to any task while excluding extraneous information and clutter. Three autonomous controllers complete the SHARED architecture, the Team Composition and Tasking (TCT)[37] module, the Team Dynamics and Tactics (TDT)[35] module and the Cooperative Path Planning (CPP)[38,39] module. Each of these three controllers handles different aspects of the overall automated aspects of mission planning and execution.

The TCT is the highest level of abstraction decision maker. Its primary responsibility is to apply the will of the commanding officers to the available knowledge of battlefield conditions in a game theoretic environment to produce a coherent overall plan. In so doing, the TCT is responsible for optimally grouping the available units into teams, assigning tasks to each team and assigning ordinance to each unit. The TCT level controller must also ensure that the tasks assigned to each team take into account both temporal and physical constraints. The next level

reasoner in the hierarchical decision tree is the Team Dynamics and Tactics level. While the TCT operates on the entire battle at once, a separate TDT is run for each team as composed by the TCT. For each unit on team, the TDT is responsible for setting waypoints, commanding sensor sweeps, specifying weapon expenditures and target selections and countermeasure deployment. To facilitate a maximum rate of success, the TDT is programmed to predict an opponent's strategy using Nash equilibrium. Adding to the overall complexity of the TDT module, in order to preserve a variable initiative command structure, special care must be taken to ensure that any individual commands issued by commanders must be followed and that commands issued by the TDT abide by the given rules of engagement (RoE). The lowest hierarchical level of detail, Cooperative Path Planning was separated into two different sections, CPPP and CPPS. The CPPP controller controls movement, optimally moving each unit from waypoint to waypoint minimizing both the time to arrival and the danger incurred along the route. In the other path planning module, the CPPS controls sensor sweep operations, maximizing the efficiency of a search over a large area.

In this chapter, we will describe an implementation of a TDT level controller which uses an ULTRA based approach to the MT-DWTA problem, as discussed previously. Our discussion will include approaches for modifying the target selection problem to include position. Among the approaches considered are the distance discount factor (DDF) and the variable time step linear movement approach. A receding horizon feedback implementation of the ULTRA based TDT controller will be presented where we will illustrate the mechanics of our model with and without feedback. We will discuss RoE adherence, including command initiative, and its relation to sensor scheduling, ensuring that targets are correctly identified and targeted without

compromising the target selection model. To protect friendly units, we will describe methods for countermeasure deployment, including RADAR jamming and decoy deployment.

5.1 INCORPERATING MOVEMENT IN THE MT-DWTA MODEL

Previously in this dissertation, we have discussed the MT-DWTA problem and a game theoretic algorithm which generates approximate solutions quickly. Recall that in a MT-DWTA scenario, teams of units simultaneously target each other, with every unit having the capacity to target every other unit. While a useful approximation in some scenarios such as ICBM warfare, in general a units position, speed and weapon range are vital characteristics that must be taken into account in any reasonable battle plan. A TDT reasoner must be capable of considering such variables when calculating a potential strategy. In this section we will consider two methods of incorporating distance into the MT-DWTA model, the distance discount factor and the variable time step linear movement approach.

5.1.1 The Distance Discount Factor

Position and vehicle speed must be considered when implementing an intelligent target assignment strategy. If neither property is taken into consideration, a strategy will often assign units to targets that are far away; an inefficient strategy that incorporates unnecessary delay and risk in the overall battle plan. Clearly, any intelligent target assignment strategy must regard closer targets as more desirable and more threatening than those farther away. One method of introducing such an incentive is the distance discount factor [40,41]. Recall that MT-DWTA objective function given in (4.13). These objective functions formed through linear

combinations of the probability of units/targets survival multiplied by the corresponding value of that unit. By discounting the value of targets that are far away we increase the probability that a unit will be assigned to a target in its vicinity. Such a method provides a natural incentive for units to attack less valuable but possibly dangerous targets on the way to more valuable objectives.

To form a mathematical definition of the distance discount factor consider the following. A single Blue UAV unit (B1) is attacking a group of three targets on the Red Team composed two Surface to Air Missile (SAM) sites represented by triangles (R2, R3) and a High Value Target HVT represented by a star (R1) as shown in Figure 5.2. Assume that the 2 dimensional locations of the Red units are given by $l_{R_1}(k)$, $l_{R_2}(k)$ and $l_{R_3}(k)$ at time k where each $l_{R_j}(k)$ is a two dimensional vector containing a longitudinal and latitudinal value for each unit j. Likewise, the position of the Blue UAV is given by a similar vector, $l_{B_1}(k)$. Each of the Euclidean distances $\{d_1, d_2, d_3\}$ shown in Figure 5.2 can then be defined as a function of the position of the Blue unit and the corresponding Red unit as follows:

$$d_1(k) = d\left(l_{B_1}(k), l_{R_1}(k)\right)$$
$$d_2(k) = d\left(l_{B_1}(k), l_{R_2}(k)\right),$$
$$d_3(k) = d\left(l_{B_1}(k), l_{R_3}(k)\right)$$

where $d\left(l_{B_i}(k), l_{R_j}(k)\right)$ is the Euclidean norm between the i^{th} Blue unit and the j^{th} Red unit at time k.

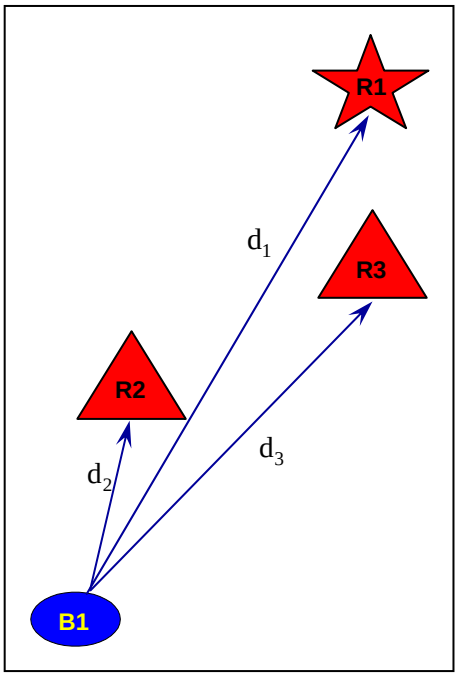

Figure 5.2 – Scenario Illustrating Distance Discount Factor

Given these assumptions and definitions we can define a distance discount factor ζ for the given Blue unit and each Red unit such that:

$$\zeta_R(j,k) = \exp\left(-\frac{d\left(l_{B_1}(k), l_{R_j}(k)\right) - \min_x\left\{d\left(l_{B_1}(k), l_{R_x}(k)\right)\right\}}{c}\right), \tag{5.1}$$

where $\min_x\left\{d\left(l_{B_1}(k), l_{R_x}(k)\right)\right\}$ is the distance between the Blue unit and the closest Red unit[5] and c is an adjustable positive constant. An example of the DDF for each Red unit is shown in Figure 5.3. Applying the expression given in (5.1) to the MT-DWTA objective functions given in (4.13) we obtain the following modified objective function for the Blue team which incorporates the distance discount factor:

[5] This value corresponds to d_2 in the scenario shown in Figure 5.2

$$J_B(u_B,u_R,k) = b_1^B B_1(k) - \sum_{j=1}^{N_R(k)} \zeta_R(j,k) r_j^B R_j(k), \tag{5.2}$$

where $\zeta_R(j,k) r_j^B$ is the reduced worth of the j^{th} Red unit.

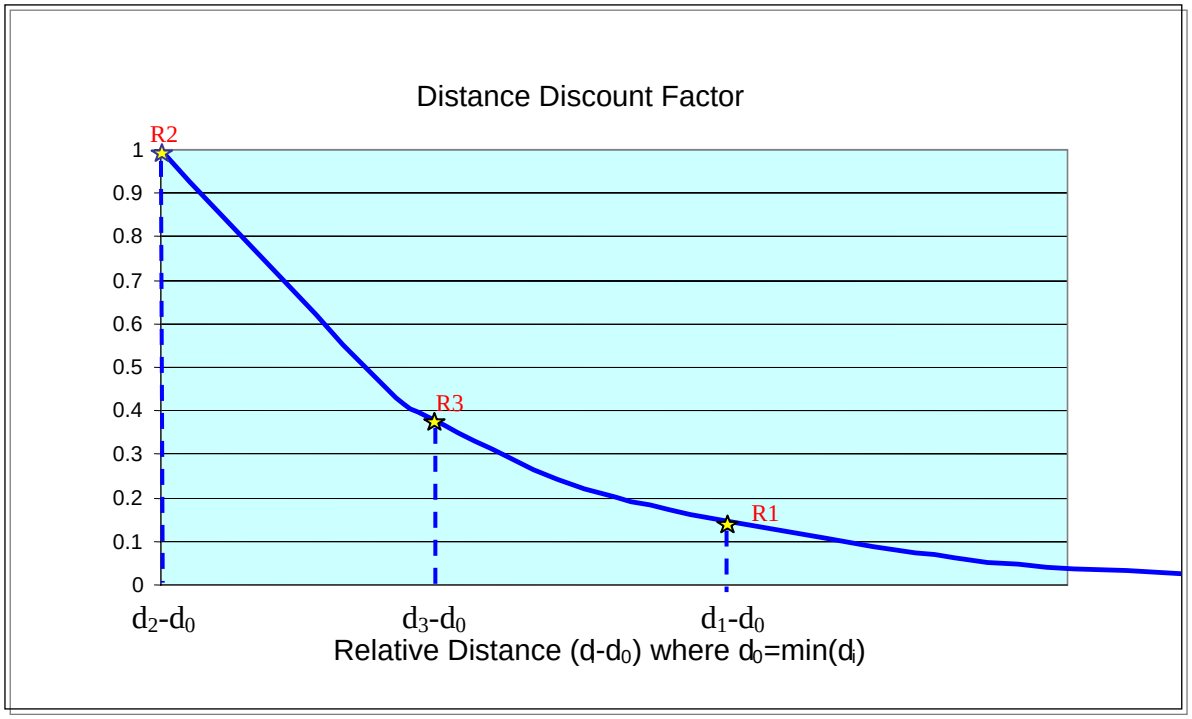

Figure 5.3 – Normalized DDF values for the Scenario Shown in Figure 5.2

Now consider a second scenario in which multiple Blue units are engaged with multiple Red units. Here the DDF definition given in (5.1) no longer holds as there exists a different distance between each of the Blue units and each of the Red units. To accommodate we will redefine the distance discount factor according geographic centre of a team's position[6]. Given N_B units on Team Blue and N_R units on Team Red, with individual unit positions $l_{B_i}(k)$ and $l_{R_j}(k)$ for

[6] While here we defined the distance discount factor according to the geographic centre of a team's position, it would have been possible to define the DDF for between each unit. Because of the combinatorial nature of the ULTRA game theoretic implementation, such an implementation is possible. However, due to the complexity involved a simpler form was implemented.

each Blue and Red unit respectively, then a median position for the Blue Team, $\bar{l}_B(k)$, and a median position for the Red Team, $\bar{l}_R(k)$, can be defined as follows, keeping in mind that each position term is a two dimensional vector containing both a latitude and a longitude term:

$$\begin{aligned} \bar{l}_B(k) &= \frac{\sum_{i=1}^{N_B} l_{B_i}(k)}{N_B} \\ \bar{l}_R(k) &= \frac{\sum_{j=1}^{N_R} l_{R_i}(k)}{N_R} \end{aligned} . \tag{5.3}$$

This is illustrated in Figure 5.4. Substituting this average position term into (5.1) yields the following expressions for the DDF from both the Blue and Red Team's perspective:

$$\begin{aligned} \zeta_R(j,k) &= \exp\left(-\frac{d\left(\bar{l}_B(k), l_{R_j}(k)\right) - \min_x \left\{ d\left(\bar{l}_B(k), l_{R_x}(k)\right)\right\}}{c} \right) \\ \zeta_B(j,k) &= \exp\left(-\frac{d\left(l_{B_i}(k), \bar{l}_R(k)\right) - \min_x \left\{ d\left(l_{B_i}(k), \bar{l}_R(k)\right)\right\}}{c} \right) \end{aligned} \tag{5.4}$$

Having an expression for both the Blue and Red Team's DDF, the overall objective functions can then be expressed as:

$$\begin{aligned} J_B(u_B, u_R, k) &= \sum_{i=1}^{N_B(k)} b_i^B B_i(k) - \sum_{j=1}^{N_R(k)} \zeta_R(j,k) r_j^B R_j(k) \\ J_R(u_B, u_R, k) &= -\sum_{i=1}^{N_B(k)} \zeta_B(j,k) b_i^R B_i(k) + \sum_{j=1}^{N_R(k)} r_j^R R_j(k) \end{aligned} . \tag{5.5}$$

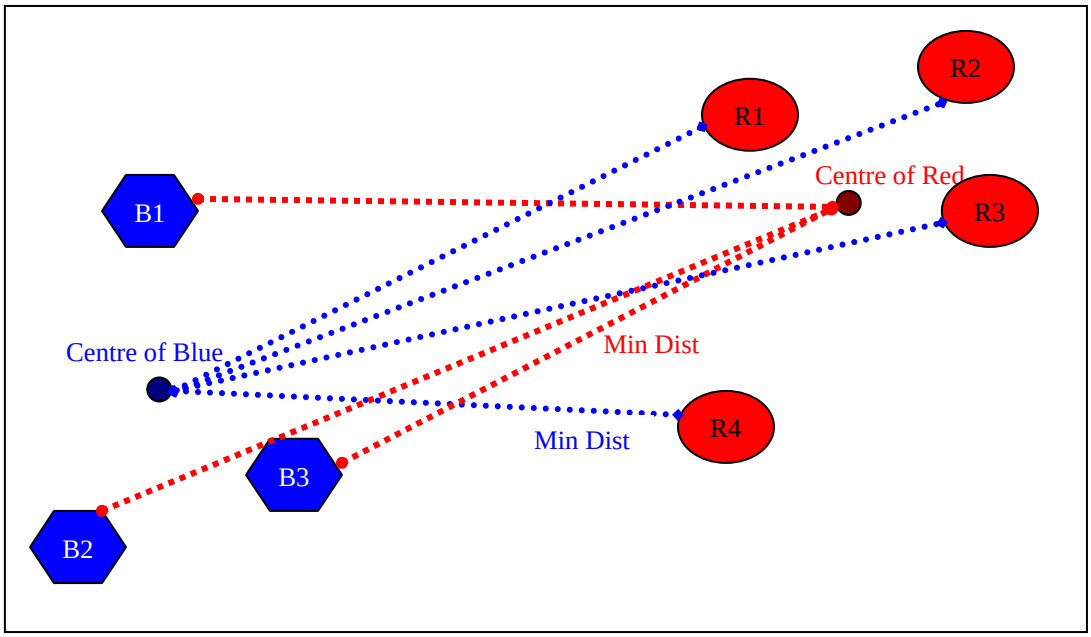

Figure 5.4 – Illustration of DDF with Team Centre

5.1.2 Experiment VII

To illustrate the effect of the DDF on a realistic scenario we will implement a MT-DWTA game theoretic approach to a scenario described in Figure 5.5 on the previously described Boing OEP. Here the Blue Team consists of a limited number of UAVs initially located at the Blue Base, labeled "Blue Area 3" in Figure 5.5. Unlike the Blue Team, the Red Team under consideration is essentially stationary[7], consisting a limited number of long range and medium range, surface-to-air missiles (SAMs) and high value Transport Erect launchers[8] (TELs) located in the area "Red Area 3". It should be noted that neither all the Blue nor all of the Red units present in Figure 5.5 are considered in Experiment VII. This is because the experiment details a possible assignment

[7] The Red Team is stationary relative to the speed of the Blue units and the size and duration of the conflict. In general some units such as Mobile SAMs can be moved, but not over the duration of the experiment.
[8] A Transport Erect Launcher is a vehicle that launches large surface to surface missiles, similar to SCUDs. Because these weapons are often used against civilian populations it is important to neutralize them as soon as possible.

from the TCT reasoner, assigning the Blue units in Blue Area 3 to the Red units in Red Area 3. The strategic objective given by the TCT to the Blue force is to neutralize the Transport Erector Launchers (TELs), which are carrying SSMs, and the integrated air defenses (IADs)[9] in Red area 3.

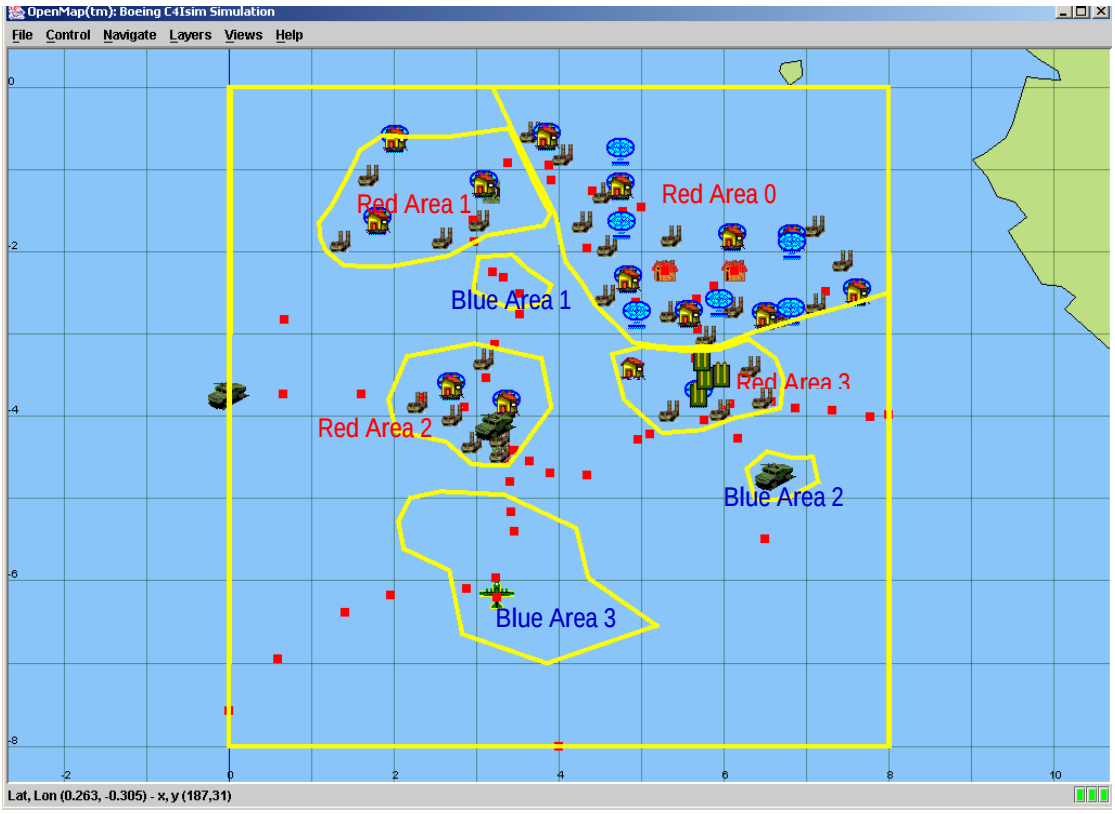

Figure 5.5 – OEP Scenario for Experiment VII

Of Red's units, the TELs are the most critical targets units as they bring most risk to the Blue base. Secondary to the main objective, Blue must destroy the defending IADs including long range and medium range SAM sites. A more detailed overview of the deployment of Red forces in Red Area 3 is shown in Figure 5.6 with a complete description including the initial equipment,

[9] IADs or Integrated Air Defenses are networks of SAM and Radar units.

the number of units, the worth of units, the weapon types and quantities for each Red unit given in Table 5.1.

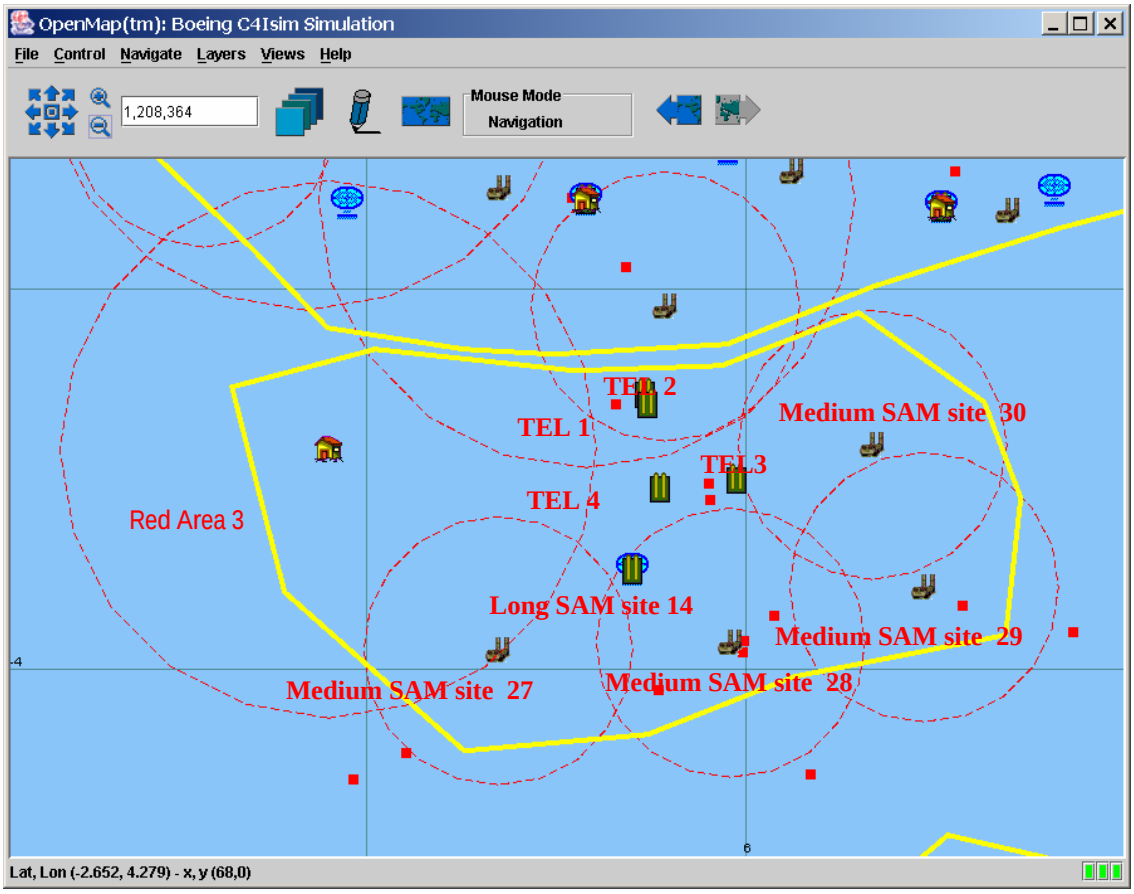

Figure 5.6 – OEP Detail of Red Area 3

It should be noted that the Red Team's SAM sites completely surround the TELs. Though the Blue units are never explicitly forced to attack their secondary targets, the SAMs, first a failure to do so will result in high loss of units.

Table 5.1 – Details of Red Units Present in Red Area 3

Red Unit (Red Area 3)	Number of Unit (total=12)	Worth of each unit	Pkill Vs UAVs	Weapon Type	Weapon Quantity Per Unit
Long Range SAM sites (1 sites)	4	10	.45	long_SAM	4
Medium Range SAM sites (4 sites)	4	7.5	.4	medium_SAM	8
Transporter Erector Launchers	4	75	.35	SSM	4

To accomplish these goals, the Blue Team has been allotted a total of 5 UAVs equipped as described in Table 5.2.

Table 5.2 - Details of Blue Units Initially Located in Blue Area 3

Blue UAVs in Team (assigned to Red Area 3)	Number of Unit (total=5)	Worth of each UAV	Weapon Type	Pkill vs Tels	Pkill vs SAMs	Weapon Quantity Per Unit
Large Weapon	1	20	seeker missile	.9	.7	20
Small Weapon	2	20	seeker missile	.9	.7	8
Small Combo	2	20	seeker missile	.9	.7	4

To calculate the controls for the case when DDF is included and when it is not in Blue's TDT reasoner we will use two separate MT-DWTA objective functions. The objective functions without DDF, i.e., $J_B(u^B, u^R, k)$ and $J_R(u^B, u^R, k)$, are given in (4.13) and the objective functions with DDF are given in (5.5). For each case, the TDT controller will calculate a two step Nash

equilibrium strategy, implementing the first step[10]. Upon the completion of the first step, the TDT controller will repeat this process, calculating a two step Nash equilibrium strategy and again implementing only the first step.

The Blue Team's TDT calculated target assignment strategies[11] for the first four battle steps are given in Table 5.3 for the case when DDF is not considered. It is not surprising that all Blue UAVs have been assigned to the most critical targets, the TELs, instead of first neutralizing the defending SAM sites. Looking at the worth of the Red Units in Table 5.1, we see that each TEL is significantly more valuable than the other Red units. Consequently, any target assignment strategy that does not incorporate movement and location will naturally assume the greatest reward comes from targeting the most valuable targets. Because the Blue Team ignores the SAM sites on their way to the primary targets all but one are destroyed during the first stage of battle. In the following rounds, having passed through Reds defensive perimeter, Small Combo 2 proceeds to select one TEL as its target until it is ultimately destroyed.

Table 5.3 - Blue Target Assignment Strategies without DDF

Blue UAV	Target Assignments (Control Output)			
	1st step	2nd step	3rd step	4th step
Large Weapon 1	TEL 4	No Target	No Target	No Target
Small Weapon 1	TEL 3	No Target	No Target	No Target
Small Weapon 2	TEL 2	No Target	No Target	No Target
Small Combo 1	TEL 1	No Target	No Target	No Target
Small Combo 2	TEL 4	TEL 3	TEL 2	No Target

[10] This Moving Horizon implementation is explained in more detail in section 5.2.

[11] Only the Blue Teams controls are given as the OEP internally provides the Red teams controls, making them essentially unknowable.

The target assignment strategies for the first four battle steps when the DDF given by (5.4) is used to reduce the relative importance of targets that are farther away is given in Table 5.4. Here we see that while Blue's primary objective remains the same, the first target assignments are to destroy the SAM sites defending the parameter of the Red Team's high value TELs. Similarly, at the second round, the Blue Team continues to weaken the defending units. In the third and fourth round, having neutralized the Red Teams IAD and having moved closer to the high value TELs, the surviving Blue units attack the primary target. Here we see an instance in which the DDF provides a substantial strategic benefit over a case where position is not considered.

Table 5.4 - Blue Target Assignment Strategies with DDF

Blue UAV	Target Assignments (Control Output)				
	1^{st} step	2^{nd} step	3^{rd} step	4^{th} step	5^{th} step
Large Weapon 1	Medium SAM Site 27	Long SAM site 14 launcher 4	TEL 3	TEL 3	No Target
Small Weapon 1	Medium SAM Site 27	No Target	No Target	No Target	No Target
Small Weapon 2	Medium SAM Site 27	Long SAM site 14 launcher 3	TEL 4	TEL 2	No Target
Small Combo 1	Medium SAM Site 27	No Target	No Target	No Target	No Target
Small Combo 2	Long SAM 14 launcher 4	Long SAM site 14 launcher 2	TEL 4	TEL 1	TEL 3

To compare the strategic benefits of the DDF, snap shots of the final outcomes after five battle steps using DDF and after four battle steps without the DDF[12] are shown in Figure 5.7 and Table 5.5. Note that with DDF, four Long-SAM-14 launchers, one medium SAM site, and four TELs are destroyed. In addition, one Blue UAV is preserved. In contrast, when the TDT reasoner fails to incorporate any position or movement details, only two TELs are destroyed and all of the Blue Teams units are destroyed. The partial outcome for the battle is given in Table 5.

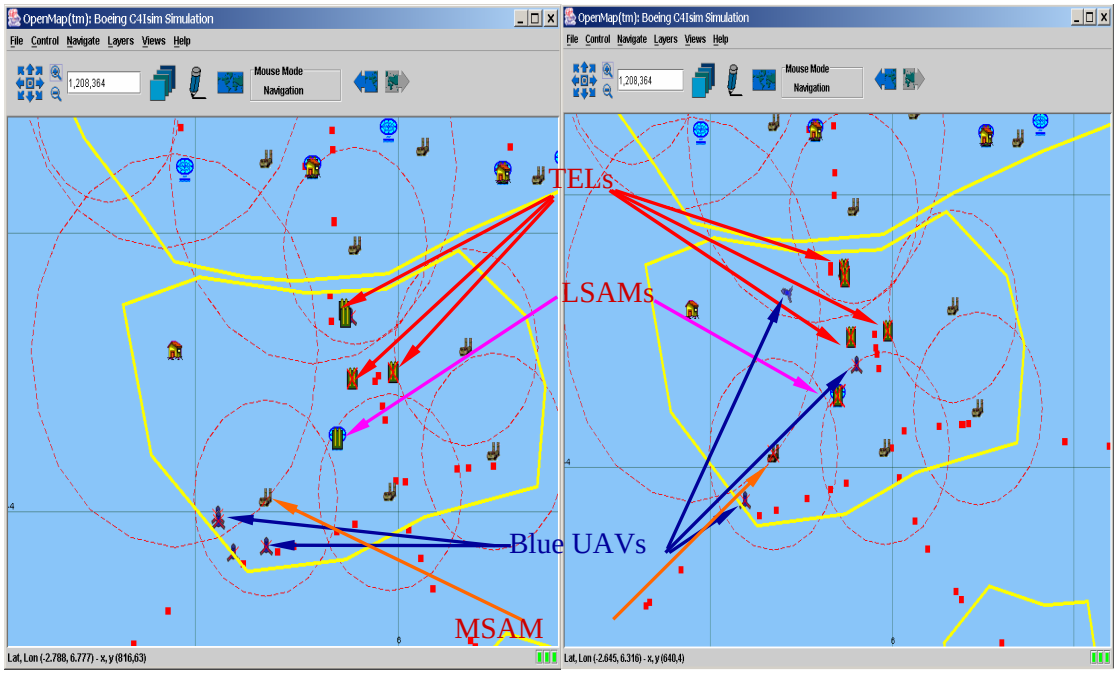

Figure 5.7 – Outcome of the Experiment VII w/ DDF (left) and without

Table 5.5 – Compilation of the Outcome of Battle in Red Area 3

Initial Number of Units	Without DDF	With DDF
4 long sam launchers	0 destroyed	4 destroyed
4 medium sam sites	0 destroyed	1 destroyed
4 TELs	2 destroyed	4 destroyed
5 UAVs	0 preserved	1 preserved

[12] The scenario in which the TDT reasoner does not incorporate the DDF cannot be evaluated over five battle steps because all the units on the Blue Team are destroyed at the end of the fourth round of targeting.

Another way to contrast the outcomes of the battle is to consider the worth's of remaining Red and Blue Teams at the end of each battle step when DDF is incorporated in the objective function and when it is excluded. Using (4.13) as a basis, the total worth of the Red and Blue force at time k can be expressed as follows:

$$W_B(k) = \sum_{i=1}^{N_B} b_i^B B_i(k) \text{ for the Blue Team at round } k \text{ and} \tag{5.6a}$$

$$W_R(k) = \sum_{j=1}^{N_R} r_j^R R_j(k) \text{ for the Red Team at round } k, \tag{5.6b}$$

where the worth values of units b_i^B and r_i^R are obtained from the third column of Table 5.1 and Table 5.2, which are also used as weighting coefficients[13] used in the objective functions (5.5). We should note that the worth of the Red and Blue teams does not include the DDF, but rather the overall worth of the team. These results are shown in Figure 5.8 and Figure 5.9, illustrating the Blue and Red Team's worth respectively as a function of time.

[13] These values are obtained from Boeing as a part of the OEP battle simulator.

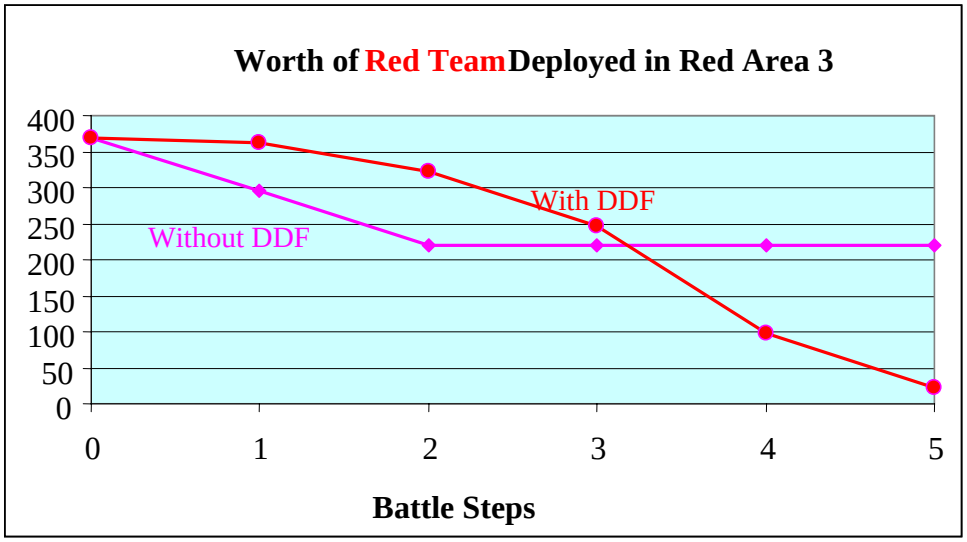

Figure 5.8 – Worth of Red Team as a Function of k w/ and w/o DDF

Note that the worth of the Red Team is initially much higher than when the DDF is used as opposed to when it is ignored. The Blue Team initially scores much lower with the DDF than without. However, more Red units are destroyed as the battle progresses when DDF is employed. When the DDF is not employed all of the Blue units are destroyed, leaving many Red units unaccounted for. This confirms our initial assessment. The TDT reasoner attacks less valuable but more dangerous targets because they are close when the DDF is incorporated in the objective functions. This weakens the Red Team's IAD at the early stages of battle, allowing more Blue units to survive and neutralize and the high value critical targets later in the battle.

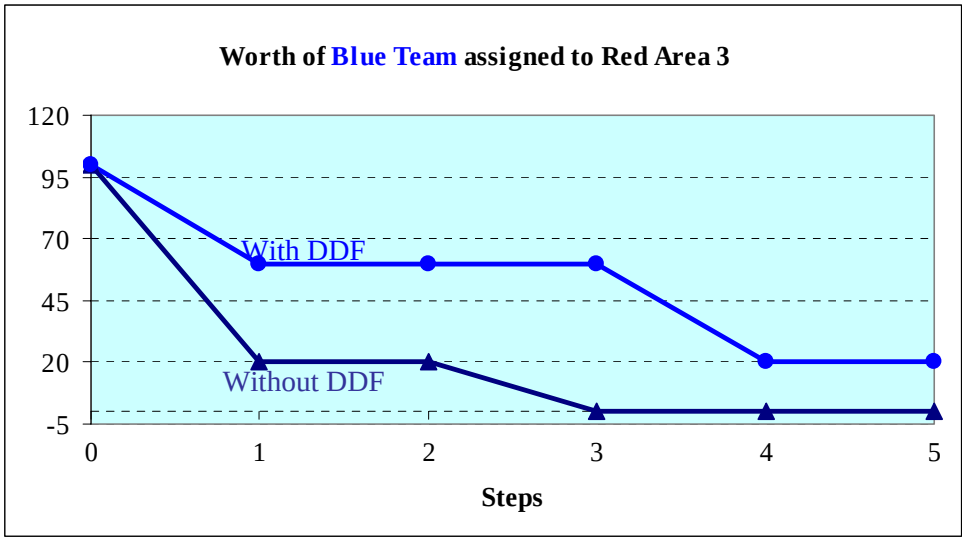

Figure 5.9 – Worth of Blue Team as a Function of k w/ and w/o DDF

Similarly, more Blue UAVs are preserved when using the feedback controller with DDF.

The plots contained in Figure 5.8 and Figure 5.9 can be combined to form another measure of performance. Consider the net performance according to the Blue Team at battle step k:

$$Net(k) = \left(W_B(k) - W_B(0)\right) - \left(W_R(k) - W_R(0)\right), \tag{5.7}$$

Essentially the total worth of the Red units destroyed minus the total worth of the Blue units destroyed. The net performance of the Blue Team is shown in Figure 5.10. As we expected, the net performance of the Blue Team is lower in the early rounds when using the DDF as opposed to not using the DDF. This performance is sufficiently negative for Team Blue that it can be said that the Blue team looses[14] for a time. We also see that because the Blue Team destroyed the Red IAD when the DDF was incorporated, the surviving Blue units are more able to destroy the Red high value targets.

[14] The Blue Team has a negative net performance value. This would correspond to the Red Team having a positive net performance value. Since the Red Team would therefore have a higher net performance value then the Blue Team, we can conclude that the Blue Team is in effect "loosing" the battle.

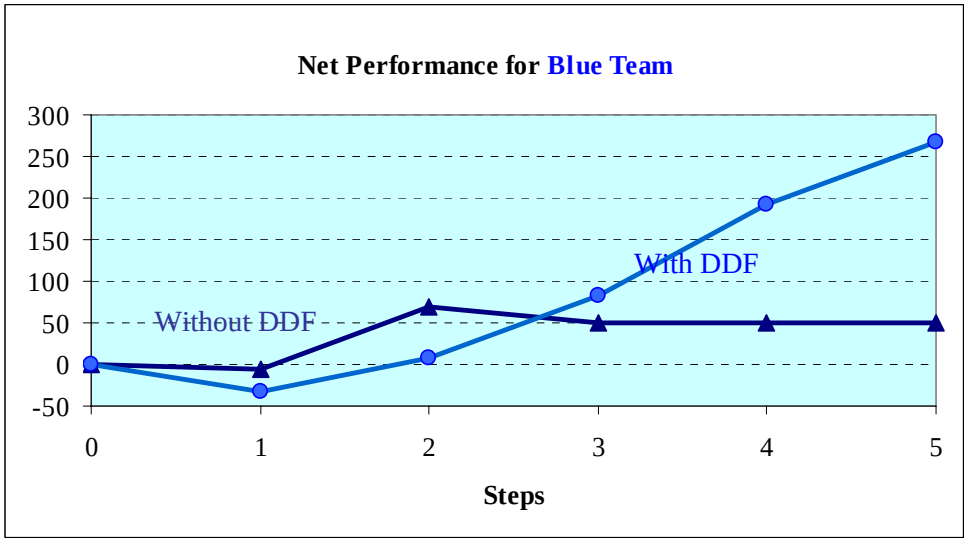

Figure 5.10 – Net Performance for the Blue Team as a Function of k w/ and w/o DDF

5.1.3 Variable Time Step Linear Movement Approach

While we have shown that the DDF is an effective method for incorporating position into a MT-DWTA model, it is not without problems. One significant problem arises when the units of a team are spread out over a large distance. In this case the distance from the centre of all units to a particular enemy unit can vary vastly from the distance of a given unit. For example, consider the scenario shown in Figure 5.11. Here we see that Blue unit 3 is very close to Red unit 3, yet Red 3 would be regarded as the most distant by the DDF. Also, we see that the Blue units are very similar in distance to the Red Team's centre meaning Red would likely assign a DDF value near 1 for each of the Blue units. The end result being that the Red Team's units would be assigned as if there was no DDF.

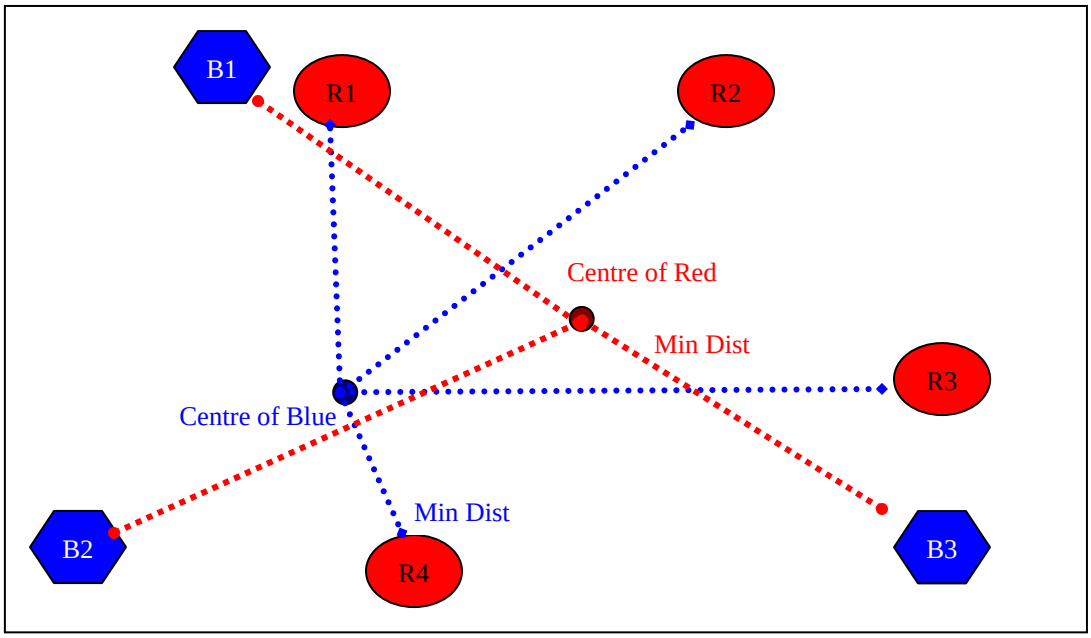

Figure 5.11 – Illustration of the DDF for Two Dispersed Teams

Another problem that arises when considering the DDF is the effects of planning a strategy over multiple battle steps. Clearly, the DDF is defined at step k, but it is difficult to determine what the DDF should be at step $k+1$ and higher. This effect becomes problematic when planning battles over multiple battle steps. While the DDF has a significant advantage over a MT-DWTA implementation that does not consider the effect of position, it is not suited to be used as the sole representation of distance when incorporating position into the MT-DWTA model. A valid implementation must not only be capable of properly penalizing targets that are far away, it must also predict where units will be after a given battle step.

To solve these problems we introduce the Variable Time Step Linear Movement approach. This method incorporates position, speed and weapon range using a rudimentary path planning algorithm. Consider a case where Team Blue, composed of N_B units is attacking Team Red, composed of N_R units. We assume that each battle step is defined over a duration of time $t(k)$.

Now say that the i^{th} unit on Team Blue located at $l_{B_i}(k)$ and having maximum velocity of $\varpi_B(i)$ is assigned to target the j^{th} unit of Team Red at the k^{th} battle step, with the j^{th} Red unit located at. This targeting will be carried out with the i^{th} Blue unit's ω^{th} weapon having a range of $\rho_{B_i}(\omega)$. Depending on these values, the i^{th} Blue unit will behave in one of three possible ways:

1. If $d\left(l_{B_i}(k), l_{R_j}(k)\right) \le \varpi_B(i)t(k)$, or the case where the i^{th} unit on Team Blue can reach its target in allotted time $t(k)$, then $l_{B_i}(k+1) = l_{R_j}(k)$. This is to say that if possible, the i^{th} unit on Team Blue will move to the location of the j^{th} unit at battle step k. Upon reaching this position, the i^{th} unit on Team Blue will launch its ω^{th} weapon upon the j^{th} unit of Team Red.
2. If $\varpi_B(i)t(k) < d\left(l_{B_i}(k), l_{R_j}(k)\right) \le \varpi_B(i)t(k) + \rho_{B_i}(\omega)$, or the case where the i^{th} unit on Team Blue can not reach its target in allotted time, but can move to within weapon range of its target, then $l_{B_i}(k+1) = \dfrac{\varpi_B(i)t(k)}{d\left(l_{B_i}(k), l_{R_j}(k)\right)}\left(l_{R_j}(k) - l_{B_i}(k)\right)$. In this case, the Blue unit will move in a direct line to its target at maximum speed for the entire duration of the battle step. At the end of the battle step, the Blue unit will launch its ω^{th} weapon at the j^{th} Red unit.
3. If $d\left(l_{B_i}(k), l_{R_j}(k)\right) > \varpi_B(i)t(k) + \rho_{B_i}(\omega)$, or the case where the i^{th} Blue unit cannot reach the j^{th} unit on Team Red nor arrive within range of its ω^{th} weapon over the duration of the k^{th} battle step, then $l_{B_i}(k+1) = \dfrac{\varpi_B(i)t(k)}{d\left(l_{B_i}(k), l_{R_j}(k)\right)}\left(l_{R_j}(k) - l_{B_i}(k)\right)$. Much

like the previous case, the Blue unit will move in a direct line to the j^{th} unit on Team Red's location at battle step k. The difference being that at the end of the battle step, the Blue unit will not launch any weapons as it is not within the necessary range.

Several facts should now be noted about the above variable time step linear movement approach. A battle step is divided into two distinct stages, a beginning and an ending. Movement occurs at the beginning of a battle step while sensing, weapon launches and battle damage calculation occurs at the end of a battle step. This has the curious effect of assuming that units launch weapons from their position at the end of a battle step and take damage from their position at the beginning of the same battle step. This can create problems when dealing with pursuit and evasion type scenarios as the pursuer will target the evader's previous location while the evader won't see any benefit from running away. However, when considering this is a TDT level reasoner, the variable time step linear movement approach does have an advantage with regards to solution stability. If a unit were to be able to change its current position, the action - reaction search would not function. Units would always move toward units that were not targeting them while moving away from units that did target them. This would have the effect of moving units out of the range of targeting units. The other side would behave similarly. The net effect would be a continuous loop of unintelligent strategies that were solely a result of modeling errors. We should also note that it is not practically to compare the DDF approach to the variable time step linear movement approach. When calculating the performance of the ULTRA algorithm, 10,000 to 20,000 runs were needed before the results stabilized enough to validate our results. However, where ULTRA is fast, capable of calculating solutions in fractions of a second, the OEP is slow, taking as much as an hour for a single run.

5.2 VARIABLE DURATION RECEDING HORIZON IMPLEMENTATION

We have previously defined the Nash equilibrium over two[15] battle step while conflicts may last over many battle steps. Recalling the earlier discussions on computational complexity, we have shown that it is not possible to consider possible target assignments for all units over every battle step due to the exponential relationship between computational complexity of the ULTRA algorithm and number of battle steps considered. Another factor that must also be taken into consideration is that battle damage assessment (BDA) information may or may not be available. Any implementation must therefore be able to operate in a full feedback, partial feedback or open loop operation. To satisfy these problems, we employ a variable duration receding horizon type implementation.

5.2.1 Receding Horizon implementation

To ensure reasonable computation times, we use a receding horizon implementation of a Nash equilibrium based TDT level reasoner. Instead of calculating the target assignments over duration of the conflict, battle steps $\{0,1,\ldots,K\}$, we calculate the target assignments for step k by finding a Nash equilibrium over the steps $\{k,k+1\}$ as defined in the definition of the Nash equilibrium in chapter 4. Having found a Nash equilibrium target assignment for the battle steps

[15] The Nash equilibrium may be defined over more than two battle steps in a MT-DWTA problem. However, it requires at least two battle steps, as mentioned in chapter 4, or the objective functions for the involved teams can be separated into a sum of objective functions, each based solely on the control of a single team.

$\{k, k+1\}$, we then assign the results of the k^{th}. We then simulate the battle through the k^{th} battle step[16], discarding the target assignments for the $(k+1)^{th}$ battle step. It should be noted that while the $(k+1)^{th}$ target assignment strategy is part of the Nash equilibrium when calculated from the k^{th} battle step, it becomes a static optimization after the target assignments have been declared for the k^{th} battle step. Consequently, to insure a non-coupled and hence relevant Nash equilibrium target assignment strategy, the strategy for each step k must be calculated for the pair k and $k+1$. To illustrate the receding horizon implementation employed in the TDT reasoner, consider the timing chart shown in Figure 5.12.

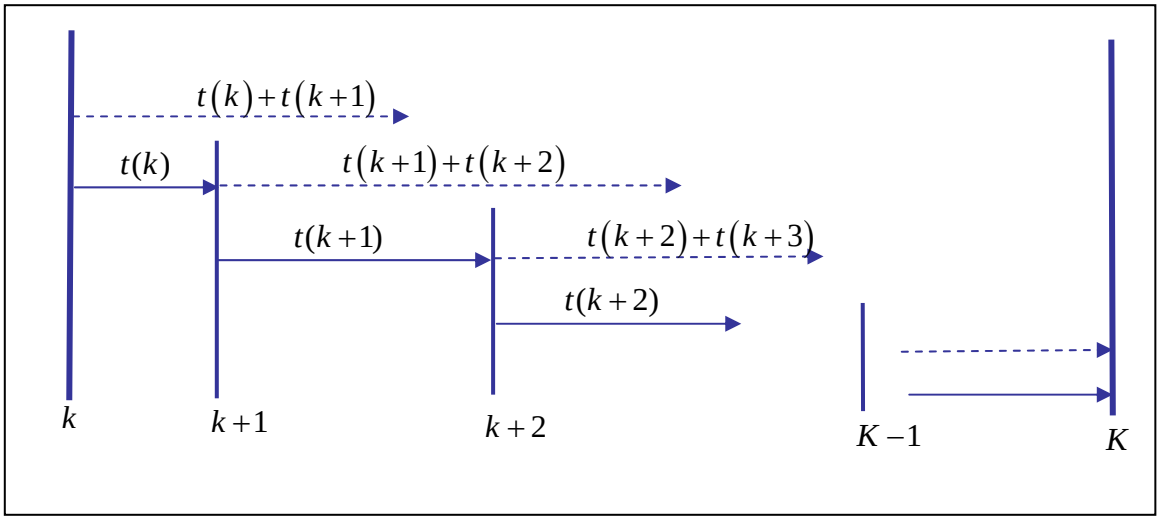

Figure 5.12 – Timing Diagram for the Variable Receding Horizon Implementation

[16] This battle simulation is carried out by the attrition model on board the ULTRA TDT controller rather than the OEP.

5.2.2 Feedback / Open- Loop Implementation

BDA feedback is another important item that must be considered when implementing a TDT reasoner. The reasoner must be capable of providing a set of tasks for each unit over the duration of the entire battle to insure unit activity in the case of communications failure. The TDT reasoner must also be capable of intelligently updating these tasks when additional information, in the form of BDA and sensor reports, is available. To accommodate this we use a state estimator based on the internal ULTRA TDT attrition model. We illustrate this controller implementation in Figure 5.13.

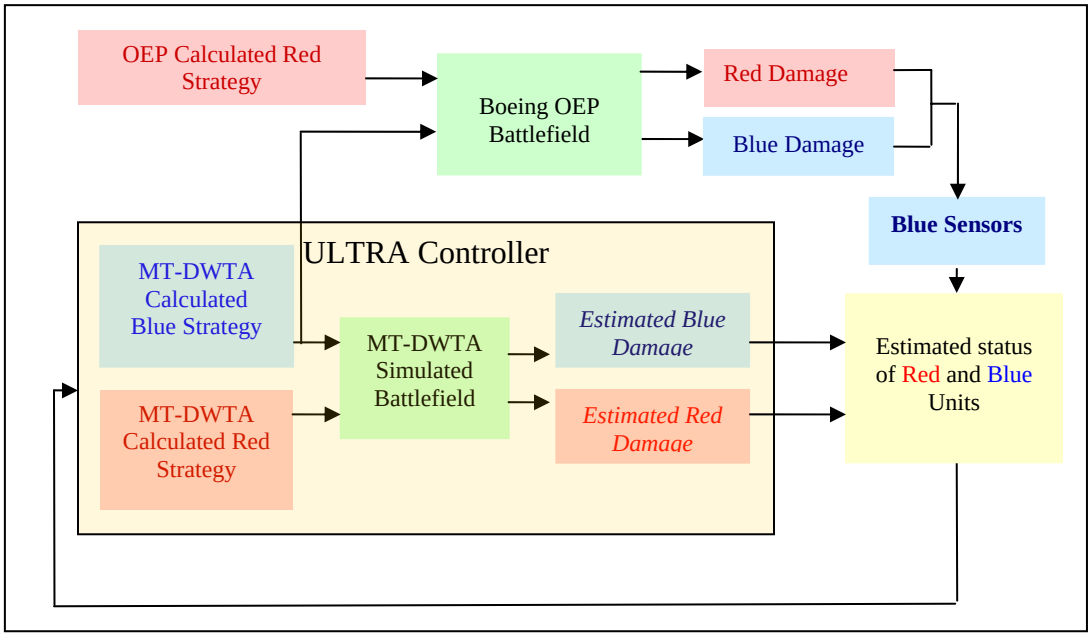

Figure 5.13 – MT-DWTA Feedback/Open-loop Implementation of a TDT Reasoner

The status of each unit at each battle step is first estimated with the MT-DWTA simulator. For example, if the i^{th} Blue unit is assigned to target the j^{th} Red unit with a weapon that has a probability of kill $p_{i,j}^{B} = .8$ at battle step k, then we assume that the j^{th} Red unit is 20% alive at

the $(k+1)^{th}$ step. This is changed when additional information is available from the blue sensor data. Say that a Blue unit then detects that the j^{th} Red unit has survived the attack. The state estimation of the Red unit is then changed from 20% to 100% and its position is updated.

5.2.3 Dynamic Battle Step Duration

An important consideration when implementing a MT-DWTA type model is the duration of the individual battle steps, $t(k)$. The duration of each battle step must balance the accuracy of the simulation versus the computational complexity of smaller battle steps. Setting the duration of a battle step too long negates the effect of incorporating distance in the MT-DWTA model. As $t(k)$ increases, a unit assumes it can target farther and farther enemy units in a single battle step. Setting the duration of a battle step too small creates a different problem, overwhelming computational complexity. Recall that an individual unit i on Team Blue can travel at a maximum velocity of $\varpi_B(i)$ and each weapon ω has a maximum range of $\rho_{B_i}(\omega)$. The minimum time to target (MTT) required for the i^{th} unit on Team Blue to move in range and target the j^{th} unit on Team read can then be calculated as follows:

$$MTT_B(i,j) = \frac{d\left(l_{B_i}(k), l_{R_j}(k)\right) - \rho_{B_i}(\omega)}{\varpi_B(i)}. \tag{5.8}$$

Recall that the ULTRA algorithm can only change F individual target assignments per iteration and only calculates the Nash equilibrium over two battle steps[17]. A units target selection strategy

[17] As defined in this dissertation, in general this may be set to any arbitrary number provided the resulting computational complexity is not overwhelming.

can only be calculated over ϕ, where $\phi = \min\{F, 2\}$. We can then say that the ULTRA algorithm will never have an incentive for the i^{th} Blue unit to target the j^{th} Red unit at battle step k if the following inequality does not hold true:

$$MTT_B(i,j) \le \sum_{\kappa=k}^{k+\phi-1} t(\kappa). \tag{5.9}$$

Defining the minimum MTT (MMTT) as the minimum time for a unit to target any unit on its adversary's team as:

$$MMTT_B(i) = \min_j \left(MTT_B(i,j) \right) \quad \text{for Team Blue and} \tag{5.10a}$$

$$MMTT_R(i) = \min_i \left(MTT_R(i,j) \right) \quad \text{for Team Red} \tag{5.10a}$$

implies that if $MMTT_B(i) > \sum_{\kappa=k}^{k+\phi-1} t(\kappa)$, then the i^{th} Blue unit will not have an incentive to target any unit. If this Blue unit is not assigned an initial target, it will choose to sit out of the battle as it cannot reach any enemy units. We can therefore say that at any battle step k,

$$\begin{aligned} MMTT_B(i) &\le \sum_{\kappa=k}^{k+\phi-1} t(\kappa) \quad \forall i \in \{1,2,\ldots,N_B\} \quad \text{and} \\ MMTT_R(j) &\le \sum_{\kappa=k}^{k+\phi-1} t(\kappa) \quad \forall j \in \{1,2,\ldots,N_R\} \end{aligned} \tag{5.11}$$

provided that each unit on Team Blue, i, and each unit on Team Red, j, are mobile[18]. As we assume that all weapons act instantaneously, the minimum time to target only accounts for the movement of a unit. Looking at (5.11), it is clear that if $t(k)$ is set too small, then a greater

[18] Mobile is relative to the speed of other units. A foot soldier can be thought of as stationary relative to the speed of a UAV. Consequently, mobility should be determined by means of a threshold of maximum velocity rather than an absolute.

degree of freedom coefficient will be required by ULTRA before the given unit will be assigned any target, greatly increasing the overall computational complexity.

The nature of an air conflict should also be taken into account. Take a typical long range bombing run for example. Here the bombers can fly for as much as 12 hours essentially towards their primary targets and as little as only a few seconds towards their secondary targets. It does not make sense to evaluate the battle in uniform battle steps. Intelligently determining $t(k)$ can greatly improve the performance of the TDT reasoner. To this end we propose a novel algorithm for determining $\{t(k), t(k+1), \ldots t(K)\}$. Assume that the conflict has reached battle step k. Here, we place all of Team Reds units in a list. We then assign $t(k) = \max_i MMTT_B(i)$, or the maximum amount of time for any unit on the Blue team to target the closest unit on the list Red units[19]. We also ensure that $t_{\min} \le t(k)K \le t_{\max}$, were $t_{\min}$ and $t_{\max}$ are user specified values. After this we search through list of Red units and remove any units that a Blue Unit can target within $t(k)$. To calculate $t(k+1)$, we repeat this operation on the reduced list. This operation is thus repeated for all $k \in \{1, 2, \ldots, K\}$.

This algorithm has several advantages and disadvantages. Consider a scenario like that given in the OEP, where both Red's and Blue's units are spatially grouped in several small clusters. Our approach has the tendency to force Blue units towards the closest cluster even if this is not optimal, especially if all Blue units travel at roughly the same speed. However, it does work well when the Blue units are significantly heterogeneous. In this case, only the slowest Blue unit is forced towards the closest Red unit. The faster Blue units then decide if it is worthwhile to attack the closest Red units or to avoid them and move towards other targets.

[19] Note that we can exclude Team Red's units from this calculation as in the OEP problem specified; all Red units are essentially stationary relative to the speed of the Blue UAVs.

5.3 ROE, SENSORS, COMMAND INITIATIVE AND COUNTERMEASURES

For a TDT reasoner to be practical, it must be capable of more than effective target selection. Many other factors must also be considered. Military conflict is intrinsically hazardous, both to the participants and to non-combatants. While a great deal of emphasis is placed on neutralizing enemy forces, commanders must also minimize collateral damage and friendly fire. For this reason military operations are governed by rules of engagement (ROE). In the military air operation specified by the OEP, the ROE are relatively simple. Essentially, no weapons may be fired without first confirming the identity of a target within a certain probability. In our model, this is governed by a probability of identity $pID_R(j)$, and a location radius of uncertainly, $locErr_R(j)$. These values represent the probability that the j^{th} Red unit has been correctly identified and the radius of uncertainty as to its current position. To determine the appropriate RoE constraints, the commander inputs a minimum probability of identity $pID_{\min}$, and a minimum location radius of uncertainty $locErr_{\min}$. Accordingly, if Blue unit i is assigned to target Red unit j, our fire control module will not fire if $\left(\left(pID_R(j) > pID_{\min}\right) or \left(locErr_R(j) > locErr_{\min}\right)\right)$.

Because of the necessity of both ROE adherence and the necessity to neutralize enemy forces, it is imperative to successfully employ sensors. Sensors are also needed to assess the effectiveness of an attack, by conducting a Battle Damage Assessment (BDA). The problem of incorporating sensing into the MT-DWTA model is complicated because we assume that all

units are heterogeneous. That is to say that the sensing abilities on all units are not uniform. For example, a reconnaissance UAV may contain many sensors but very few if any weapons while a heavy attack UAV may contain many weapons but little to no sensing ability. This prevents a simple approach, where UAVs are assigned to aim their sensors at units before and after they attack. To incorporate sensing, we introduced additional objective function values. In the standard MT-DWTA model each unit is assigned two objective function values, for example the worth of a Blue unit to Team Blue and the worth of that same unit to Team Red. We extended this to add a three more terms, a benefit for identifying an enemy unit, a benefit for locating an enemy unit, and a benefit for conducting a BDA. When a UAV targets an enemy unit that is either not identified, not located or has recently been targeted and has not yet undergone a BDA, the TDT reasoner schedules a sensor run for that battle step.

The TDT reasoner also has other responsibilities in addition to its own task scheduling algorithms. The reasoner must be a mixed initiative controller. That is to say that a commander must be capable of controlling any aspect of the final mission plan. If a commander wants a given UAV to target a certain target, then that targeting must be enforced regardless of its optimality. The ULTRA algorithm was designed with mixed initiative control in mind. Recall that it is a neighborhood search algorithm, finding more optimal strategies by changing individual target assignments. All that is needed to incorporate a mixed initiative control is to allow a commander to fix certain target assignments. In this way, an ULTRA based MT-DWTA implementation can provide a commander with an optimal target assignment strategy given a set of fixed target assignments.

Finally, countermeasures are also an important part of a coherent strategy. The intelligent use of radar jamming and decoys can provide a significant advantage to a team in terms of the

survivability of its units. In our TDT reasoner, we make use of a simple jamming and countermeasure strategy. We assume that units will jam any unit they target. To illustrate our implementation of the effect of jamming, consider the following situation. If the i^{th} unit on Team Blue is assigned to target the j^{th} unit on Team Red and the i^{th} Blue unit has jamming ability, then the effectiveness of the j^{th} Red unit against the i^{th} Blue unit will be reduced. This is to say that the probability of kill, $p^R_{j,i}$ will be reduced by some number. In our case, we arbitrarily reduced $p^R_{j,i}$ by ½ [20]. Decoy controls are calculated in an equally simple manner. If a unit possesses a decoy and we predict that it will be targeted by a Red unit, the TDT reasoner will launch a decoy at the attacking Red unit.

5.3.1 Experiment VIII

To illustrate the effectiveness of our countermeasure implementation, consider the following scenario. Assume that the Blue Team is engaged with the Red Team in a conflict simulated by the Boeing OEP. Here the Red Team is composed of 38 units of 7 types as given in Table 5.6 while the Blue Team is composed of 11 units of 3 types as given in Table 5.7.

[20] This number is selected arbitrarily to simulate some of the effect of jamming. In general, this value would have to be generated empirically in the same manner as the probability of kill matrices.

Table 5.6 – Composition of Team Red for Experiment VIII

Red Unit (in Red Area 2)	# in Unit (total=38)	Worth of each unit* (refer to Boeing OEP)
Long Range SAM Sites (2 sites)	8	10
Medium Range SAM Sites (6 sites)	6	7.5
Mobile SAM Sites	4	12
SPARTY	4	10
Personnel Carriers	5	10
Communication Vans	1	10
Tanks	10	10

Table 5.7 – Composition of the Team Red for Experiment VIII

Blue UAVs TEAM 1 (assigned to Red Area 2)	# in Unit (total=11)	Worth of each UAV* (refer to Boeing OEP)	Decoys
Large Weapon	3	20	5
Small Weapon	4	20	5
Small Combo	4	20	5

We define the probability of kill matrices to be the same as given in Experiment VII with the Blue units having the same level of effectiveness against Tanks, SPARTYs, APCs and Comm Vans as previously given to Tels. Finally, the Red Team is assumed to be deployed as shown in Figure 5.14.

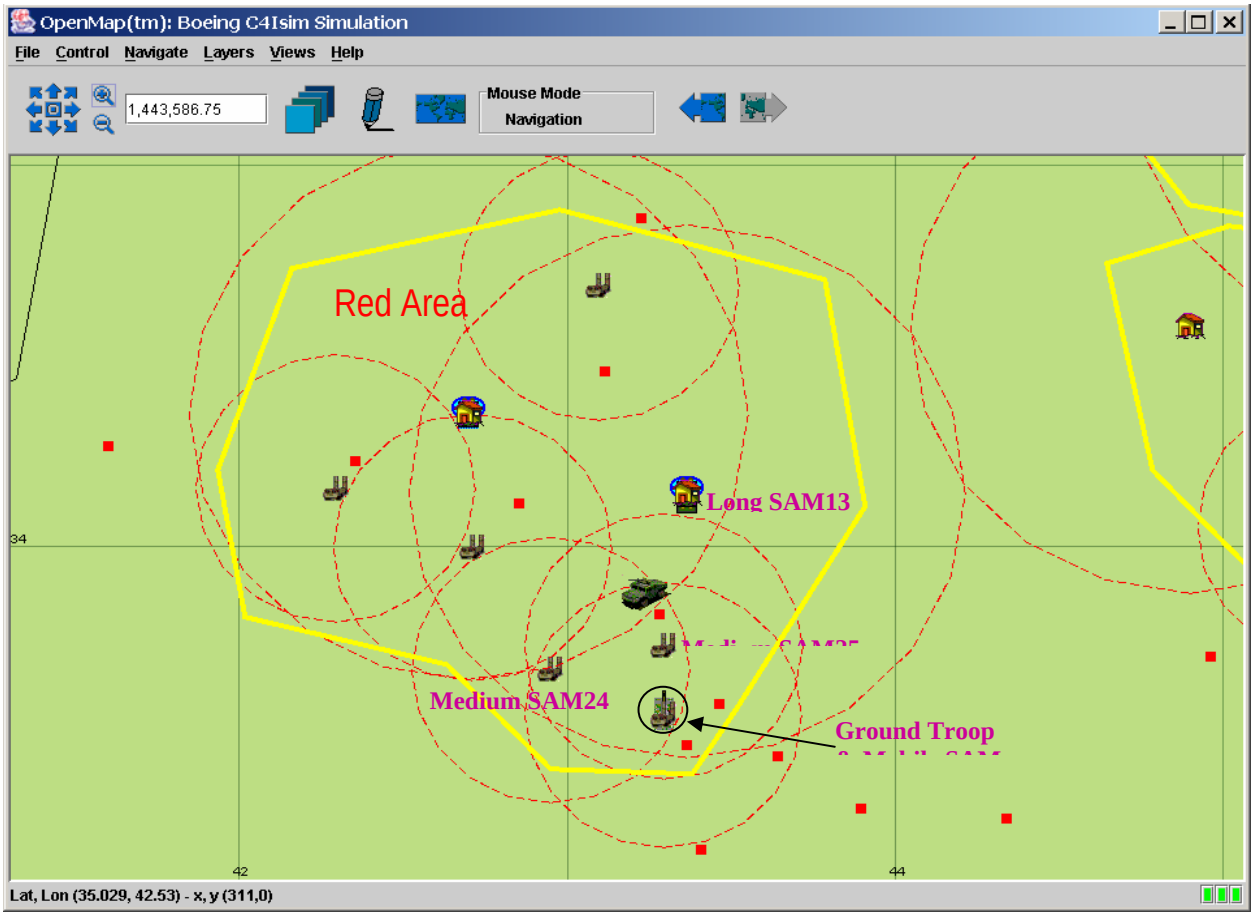

Figure 5.14 – Deployment of Team Red in Experiment VIII

In this experiment, we evaluated the performance of four separate countermeasure controls using the ULTRA MT-DWTA TDT reasoner; no countermeasures, jamming only, decoy only and a combination of jamming and decoy. Due to the temporal constraints, each of these experiments was only run a single time on the Boeing OEP. The results of this experiment are shown in Table 5.8. Here we see that a countermeasure controller can greatly improve the overall performance of the TDT controller. We should also note that even though the TDT reasoner performed worse when both jamming and decoy controls were employed than just decoy alone, this is not reflective of actual performance. The OEP platform is dependent on Monte' Carlo

type evaluations to determine weapon hits and misses. As such, running the experiment with a different random seed can have a moderate effect on the overall result of the battle.

Table 5.8 – Experiment VIII Results

Unit Names	Initial Forces	No Jamming or Decoy	Jamming Only	Decoy Only	Jamming and Decoy
Mobile SAM Sites	4	4	1	1	3
Ground Forces	20	17	16	14	13
Large Weapon	3	0	2	2	2
Medium Weapon	4	0	0	3	3
Small Weapon	4	0	0	2	1

6.0 CONCLUSION

Much of modern military thought is aimed at increasing the efficiency of current weapon systems and decreasing the risk to battle participants. As a result, the control of unmanned vehicles continues to be a main emphasis of current military research. One method of improving the performance of unmanned vehicles is with the use of automated controllers. These controllers are designed to either replace or assist the commander in battle planning and the control of unmanned vehicles. To accomplish this, controllers model the battle space, generate an objective function to metric a team's performance and then employ an algorithm to optimize this objective. Often in these models, the outcome of the battle is dependent on the strategy of more than a single decision maker. Although significant, this coupling is typically ignored in favor of simple naïve controllers as it is not possible to incorporate such interdependencies in standard optimization techniques. These types of competitive problems are best handled through the use of Game Theory. However, traditional game theoretic methods are often computationally intractable, even for scenarios with small numbers of units.

In this dissertation, our starting point is the standard weapon target assignment problem in which a single team of units is to be assigned to a set of targets. While such model is appropriate for scenarios such as ICBM warfare in which a target assignment strategy can be assumed independent of the possible actions of the enemy, it is not well suited for scenarios that include an adversarial force. To more realistically model combat, we extended this model to account for multiple teams of units targeting other teams of units. Here, combat is modeled with the

rationale that each team has certain information regarding its own and its adversary's units. Any intelligent team target assignment strategy must therefore aim not just to destroy that team's enemy's units, but also to preserve its own. Thus, each team must take into account the possible target assignment strategies of its adversaries. Game theoretic solution concepts, the Nash equilibrium in particular, have proven to be effective in solving these kinds of competitive problems. Typically, the game theoretic problems are solved by constructing a game matrix, with each index containing all of the possible strategies of a single decision maker and each entry containing the objective function values for each team given the corresponding strategies. We showed that this game matrix approach becomes computationally intractable, even for small instances of the MT-DWTA. For the MT-DWTA to be an effective model for military conflicts, an algorithm is required to quickly find Nash or near Nash equilibrium strategies.

To solve the MT-DWTA we first examine a simple case (SMT-DWTA), one in which a team knows its adversary's target assignment strategies a priori. Here because a team knows the target assignment strategies of its adversaries, the effects of these strategies can essentially be removed from that team's objective function. This allows the problem to be solved through more conventional optimization methods. To allow for maximum flexibility for implementing features such as command initiative, we use a large scale neighborhood search algorithm which we denote ULTRA. In a large scale neighborhood search algorithm (LSNS), an initial strategy is chosen. A neighborhood of similar strategies is then formed around this initial strategy. A LSNS algorithm then finds a strategy in this neighborhood that is better than the initial strategy. Having completed the first step, a new neighborhood is generated around the second strategy. This continues until a strategy is optimal in its own neighborhood. In the case of ULTRA, we assume that this neighborhood contains all target assignment strategies differing from the initial

target assignment strategy by no more than F units, where F is the degree of freedom coefficient. Because ULTRA yields a near optimal solution, we performed a series of experiments to determine its performance under various conditions. We found that the best strategy to initialize the ULTRA algorithm is the unit greedy strategy, in which each unit is assigned to its optimal target independent of the other units of its team. Using this initial strategy, we show that the ULTRA algorithm will generate a target assignment strategy that is on average 95% optimal when $F = 1$. We also show that in this case, the ULTRA algorithm will perform better than 90% optimal more than 90% of the time and will perform no worse than approximately 75% optimal for large numbers of units.

Using the ULTRA algorithm, we are then able to efficiently solve the MT-DWTA in a two team problem numerically using the standard action - reaction search. This search algorithm functions by iteratively calculating the optimal reaction of one team to a target assignment strategy of a second team and then in turn calculates the optimal target assignment strategy of the second to that target assignment strategy of the first. We assume that a Nash equilibrium is found when neither team changes their target assignment strategies. This method has the distinct advantage in that it will converge even if there is no Nash equilibrium. While not immediately apparent, we argue that this algorithm will converge to strategies that are near Nash equilibrium when the Nash equilibrium does not exist. We defend this concept by expanding the definition of the Nash equilibrium. To prove that our target assignment method generates viable strategies, we consider experimental cases in which teams employ strategies other than the Nash equilibrium. We compare four different strategies, the unit random, unit greedy, team optimal and the Nash equilibrium. We demonstrate that while our method produces strategies that are not guaranteed to be optimal in the traditional sense, the Nash equilibrium yields objective function

values that are consistently higher than any other strategy considered, regardless of the strategy employed by the adversary. This is significant as a non-optimal strategy that takes into account the adversary's strategy has been shown to perform better than an optimal team target assignment strategy that does not take into account the possible target assignments of its adversary. Furthermore, we show that this effect can be quite marked, depending on the nature of the scenario. Cases where the units on each team are significantly heterogeneous while the overall effectiveness of the aforementioned teams is balanced yield the largest difference between the team optimal and Nash equilibrium target assignment strategies. In contrast, unbalanced scenarios yield little to no difference between the Nash equilibrium and the team optimal approach.

Having generated a game theoretic combat model and presented an algorithm capable of efficiently generating solutions, we put forward a design for the controller at the Team Dynamics and Tactics level of a mixed initiative battlefield management system called SHARED. After introducing the hierarchy of control, we begin to examine what is needed to realistically model a battle using the MT-DWTA. We first examine the issue of position and movement. Because the MT-DWTA makes a basic assumption that all units are capable of targeting all other units at any battle step, it can produce target assignments that pay little heed to the temporal constraints incurred when a target is far away. To compensate for this we introduce two methods. The first method, denoted distance discount factor (DDF), operates by reducing the value of targets that are far away. This provides an incentive for units to target adversarial units that are nearby and thus more accessible. While this method is effective, creating better performing targets assignment strategies then the MT-DWTA alone, the DDF does have several drawbacks that leave it incomplete. We then introduce a second method to solve some of these shortcomings

using a rudimentary path planning algorithm to account for position and movement. We also present methods for generating battle step durations, creating battle plans with and without feedback, accounting for command initiative, abiding by rules of engagement, assigning sensors and deploying countermeasures.

6.1 FUTURE WORK

We have presented a complete model that is useful for designing military command controllers; however this subject is by no means complete. While the ULTRA algorithm is fast, it too becomes computationally intractable for very large systems. This is especially true of instances in which large numbers of heuristics are required to evaluate the objective function values, as in the case of a TDT controller. Another problem lies in the uncertainty which permeates the entire model. The probability of kills matrices are seldom known exactly, the worth of friendly units and enemy units are generally arbitrarily valued, the values that an enemy unit ascribes to the units are assumed and subject to error, there is no guarantee that the opponent is using the same model and the fog of war all contribute to an extremely high level of uncertainty. While we did show that Nash equilibrium type strategies can show a considerable advantage over naïve approaches in our model, the implicit uncertainty in the model combined with errors between the model and reality could offset any such improvement. As such, there is a large body of work to be done in this field.

One promising approach to account for the uncertainty inherent in a conflict is through the use of ordinal games [42, 43]. Ordinal games are games that do not use payoff functions. Instead the objective information is represented through a preferential ranking of all possible outcomes. Instead of commanders arbitrarily fixing objective function coefficients, a

commander would subjectively rank a series of outcomes. This may alleviate some of the uncertainty caused by a commander arbitrarily selecting objective function coefficients. However, certain obstacles must be overcome to make an ordinal approach feasible. The largest drawback to an ordinal approach is that a commander must rank all possible outcomes, a substantial feasibility issue for MT-DWTA type problems. One possible solution is to build an automated expert system that could be trained to coarsely approximate the opinion of a commander, greatly reducing the actual number of comparisons required of the commander.

Another method that could be used to remove an amount of uncertainty in the model and reduce the overall computational complexity is to switch from a target assignment based control structure to a network flow based approach [44]. Networks provide a very powerful system of techniques for dealing with combinatorial optimization problems. A network flow is a system of nodes connected by arcs with "flow" traveling from source nodes to sink nodes. More powerful than linear programming, networks allow minimum and maximum flow constraints and separate cost per unit flow to be placed on arcs. However, while some work has been done regarding network games [45], more work is required to intelligently model competitive teams of heterogeneous units with distinct capabilities and a common goal.

In this dissertation, we have confined the application of the MT-DWTA to teams of UAVs and teams of ground based units. This has been done to promote clarity of concept and does not infer any lack of generality of the MT-DWTA model. One promising application of mixed initiative battlefield controllers is as a part of the United States military's future combat system (FCS). Instead of providing guidance to commanders planning military air campaigns, such controllers may prove to be an invaluable asset to infantry engaged in urban combat. Using game theoretic techniques, it is hoped that an intelligent system will be capable of determining

probable ambush locations, suggest possible plans of attack, optimally re-supply troops and retrieve casualties with the optimal balance of speed and risk avoidance. As there is no theoretical reason why the techniques for generating a MT-DWTA strategy discussed in this dissertation would not apply to ground rather than air based combat, another avenue of future research is to modify the TDT controller described in Chapter 5 to operate on a ground based combat environment. With research being funded by groups such as the DARPA RAID project, automated battle planners should continue to be a major emphasis of research and development for years to come.

BIBLIOGRAPHY

[1] Manne, Alan S., "A Target Assignment Problem", *Operations Research*, pp. 346-351, 1958

[2] Lloyd, S. and H. Witsenhausen, "Weapons Allocation Is NP-Complete," In *IEEE Summer Simulation Conference*, 1986

[3] Day, R.H., "Allocating Weapons to Target Complexes by Means of Nonlinear Programming," *Operations Research* Vol. 14, pp. 992-1013, 1966

[4] Wachholder, E., "A Neural Network-Based Optimization algorithm for the Static Weapon-Target Assignment Problem," *ORSA Journal on Computing*, Vol. 4, pp. 232-246, 1989

[5] Lee, Z-J. Lee, Su, S-G, and Lee C-Y, "Efficiently Solving General Weapon-Target Assignment Problem by Genetic Algorithms with Greedy Eugenics," *IEEE Transactions on Systems Man and Cybernetics Part B*, Vol. 33, No. 1, pp. 113-121, 2003

[6] denBroeder, G.G. Jr., R.E. Ellison, and L. Emerling, "On Optimal Target Assignments," *Operations Research* Vol**. 7**, pp. 322-326, 1959

[7] Hosein, P., A *Class of Dynamic NonLinear Resource Allocation Problems*, PhD Thesis, Massachusetts Institute of Technology, MA, 1989

[8] Hosein, P., J. Walton and M. Athans, "Dynamic Weapon-Target Assignment Problems with Vulnerable C3 Nodes," In *Proceedings of the 1988 Command and Control Symposium*, June 1988

[9] Murphy, R.A. "Target-Based Weapon Target Assignment Problems," *Nonlinear Assignment Problems: Algorithms and Applications*, Vol. 7, P.M. Pardalos and L.S. Pitsoulis (Eds.), Kluwer Academic Publishers, pp. 39-53, 1999

[10] Matlin, S.M. "A Review of the Literature on the Missile-Allocation Problem," *Operations Research*, Vol. 18, pp. 334-373, 1970

[11] Eckler, A.R., and S.A. Burr, *Mathematical Models of Target Coverage and Missile Allocation*, Military Operations Research Society, Alexandria, VA. 1972

[12] Voss, S. "Heuristics For Nonlinear Assignment Problems" *Nonlinear Assignment Problems: Algorithms and Applications*, Vol. 7, P.M. Pardalos and L.S. Pitsoulis (Eds.), Kluwer Academic Publishers, pp. 175-215, 1999

[13] Nash, J.F, "Equilibrium Points in N-Person Games," *Proc. of the Nat. Academy of Sciences*, Vol. 36, pp. 48-49, 1950

[14] Starr A.W., and Ho, Y.C., "Nonzero-Sum Differential Games," *Journal of Optimization Theory and Applications*, Vol.3, No.3, pp.184-206, 1969

[15] J. B. Cruz, Jr., M. A. Simaan, A. Gacic, H. Jiang, B. Letellier, M. Li, and Y. Liu "Game-Theoretic Modeling and Control of Military Operations," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 37, No. 4, pp. 1393-1405, 2001

[16] J. B. Cruz, Jr., M. A. Simaan, A. Gacic, and Y. Liu "Moving Horizon Nash Strategies for a Military Air Operation," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 38, No. 3, pp. 989-999**,** 2002

[17] Y. Liu, M. A .Simaan, and J. B. Cruz, Jr. "An Application of Dynamic Nash Task Assignment Strategies to Multi-Team Military Air Operations," *Automatica*, Vol. 39, pp.1469-1478, 2003

[18] Osborne, M. J., *An Introduction to Game Theory*, Oxford University Pres, New York, 2004

[19] Vives, X., *Oligopoly Pricing: Old Ideas and New Tools*, The MIT Press, Cambridge, MA, 2000

[20] Von Stackelberg, Heinrich, *The Theory of the Market Economy*, Alan T. Peacock, trans., Oxford University Press, New York, 1952

[21] Von Neumann, John, "Zur Theorie der Gesellshaftsspiele," *Mathematische Annalen*, Vol 100, pp. 295-320, 1928; translated by Sonya Bargmann in A. W. Tucker and R.D. Luce, eds., Contributions to the Theory of Games, Volume IV, Annals of Mathematics Study Vol. 40, Princeton University Press, Princeton, New Jersey, 1959.

[22] Von Neumann, John and O. Morgenstern, *Theory of Games and Economic Behavior*, Princeton University Press, Princeton New Jersey, 1944

[23] Simaan, M. A. and J. B. Cruz Jr, "On the Stackelberg Strategy in Nonzero-Sum Games," *Journal of Optimization Theory and Applications*, Vol. 11, No. 5, May 1973, pp. 533-555.

[24] Simaan, M. A. and J. B. Cruz Jr, "Additional Aspects of the Stackelberg Strategy in Nonzero-Sum Games," *Journal of Optimization Theory and Applications*, Vol. 11, No. 6, June 1973, pp. 613-626.

[25] Simaan, M. A. and J. B. Cruz Jr, "A Stackelberg Solution for Games with Many Players," *IEEE Transactions on Automatic Control*, Vol. AC-18, No. 3, June 1973, pp 322-24.

[26] Ahuja, R.K., O. Ergun, J.B. Orlin, and A.P. Punnen, "A Survey of Very Large-Scale Neighborhood Search Techniques," *Discrete Applied Mathematics*, Vol. 123, pp. 75-83, 2002

[27] Rasch, R., A. Kott and K. Forbus, "Incorporating AI into Military Decision Making: an Experiment," *IEEE Intelligent Systems*, Vol. 18, No. 4, pp. 18-26.

[28] Cruz, J. B. Jr., M. A. Simaan, A. Gacic and Y. Liu, "Moving Horizon Game Theoretic Approaches for Control Strategies in a Military Operation," *40th IEEE Conference on Decision and Control*, Orlando FL, December 4-7, 2001.

[29] Cruz, J. B. Jr., M. A. Simaan, A. Gacic and Y. Liu, "Moving Horizon Nash Solution for Dynamic Games with Application to Militariy Operations," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 38., No. 3, July 2002, pp. 989-999.

[30] Liu, Y., M. A. Simaan and J. B. Cruz Jr., "An Application of Dynamic Nash Task Reassignment Strategies to Multi-Teams Military Air Operations," *AUTOMATICA-Journal of the International Federation of Automatic Control*, Vol 39. No. 8, August 2003, pp. 1469-1478.

[31] Liu, Y., M. A. Simaan and J. B. Cruz Jr., "Game Theoretic Approach to Cooperative Teaming and Tasking in the Presence of an Adversary," *Proceedings of the 2003 American Control Conference*, Denver Co, June 4-6, 2003, pp5375-5380.

[32] Strategies for Human-Automaton Resource Entity Deployment, Special Invited Session (ThAPI), *Proceedings of the 42nd IEEE CDC*, Hawaii, December 2003, pp. 3549-3590.

[33] Liu, Y., D. Galati and M. A. Simaan, "Team Dynamics and Tactics in SHARED," *42nd IEEE Conference on Decision and Control*, Maui HI, December 9-12, 2003, pp. 4084-4089.

[34] Simaan, M. A., J. B. Cruz Jr., Y. Liu and D. Galati, "Task Assignment for Cooperative Teams in Competitive Systems," *8th World Multi-Conference on Systemics, Cybernetics and Informatics*, Orlando FL, July 18-21, 2004, Vol. XVI, pp 324-329.

[35] Liu, Y., D. Galati and M. A. Simaan, "A Game Theoretic Approach to Team Dynamics and Tactics in Mixed Initiative Control of Automa-Teams," *43rd IEEE Conference on Decision and Control*, Paradise Island Bahamas, December 14-17, 2004.

[36] Penner, R. R. and E. S. Steinmetz, "Automated interaction design for command and control of military situations," *Proceedings of the 9th international conference on Intelligent user interface*, Funchal, Maderia, Portugal, 2004, pp 362-363.

[37] Xu, L. and U. Ozguner, "Battle management for unmanned aerial vehicles", *42nd IEEE Conference on Decision and Control*, Maui HI, December 9-12, 2003pp. vol. 4, 3585-3590

[38] Ganapathy, S. and K. M. Passino, "Agreement strategies for cooperative control of uninhabited autonomous vehicles," *Proceedings of the 2003 American Control Conference,* Denver Co, June 4-6, 2003, Vol. 2. pp. 1026-1031.

[39] Flint, M., E. Fernandez and M. Polycarpou, "Stochastic Models of a Cooperative Autonomous UAV Search Problem", *Military Operations Research Journal* Vol 8. No. 4. 2003

[40] Liu, Y., "Nash-Based Strategies for the Control of Extended Complex Systems", *Ph.D. Dissertation,* University of Pittsburgh, 2003.

[41] Liu, Y., D. Galati and M. A. Simaan, "Nash Strategies with Distance Discount Factor in Target Selection Problems," *2004 American Control Conference,* Boston MA, June 29-July2, 2004, pp. 2356-2361.

[42] Cruz, J. B. Jr. and M. A. Simaan, "Ordinal Game Theory – A New Framework for Games without Payoff Functions," *2002 Mediterranean Conference on Control and Automation,* Lisbon, Portugal, July 9-12, 2002.

[43] Simaan, M. A. and J. B. Cruz Jr., "An Ordinal Approach for Decision Making in a Competitive Environment," *Hawaii International Conference on Business,* Honolulu, HI, June 18-22, 2002.

[44] Ahuja, R. K., T. L. Magnanti and J. B. Orlin, *Network Flows*, Prentice-Hall Inc., Englewood Cliffs, NJ, 1993.

[45] Castelli, L., G. Longo, R. Pesenti and W. Ukovich, "Two-Player Noncooperative Games over a Freight Transportation Network," *Transportation Science*, Vol. 38, No. 2, May 2004, pp. 149-159.