2006

A Comparison of Agent-Based Optimization Approaches Applied to the Weapons to Targets Assignment Planning Problem

Hitesh Deepak Soneji
Wright State University

Follow this and additional works at: https://corescholar.libraries.wright.edu/etd_all

Part of the Engineering Commons

Repository Citation
https://corescholar.libraries.wright.edu/etd_all/40

This Thesis is brought to you for free and open access by the Theses and Dissertations at CORE Scholar. It has been accepted for inclusion in Browse all Theses and Dissertations by an authorized administrator of CORE Scholar. For more information, please contact corescholar@www.libraries.wright.edu, library-corescholar@wright.edu.
A Comparison of Agent-Based Optimization Approaches Applied to the Weapons to Targets Assignment Planning Problem

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Engineering

by

SONEJI HITESH DEEPAK
B.E., Government College of Engineering, Pune, India, 2002

2006
Wright State University
WRIGHT STATE UNIVERSITY  
SCHOOL OF GRADUATE STUDIES  

August 21, 2006

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Soneji Hitesh Deepak ENTITLED A Comparison of Agent-Based Optimization Approaches Applied to the Weapons to Targets Assignment Planning Problem BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science in Engineering.

__________________________________________  
Raymond R. Hill, Ph.D.  
Thesis Director

__________________________________________  
S. Narayanan, Ph.D., P.E.  
Department Chair

Committee on  
Final Examination

__________________________________________  
Raymond R. Hill, Ph.D.

__________________________________________  
Frank W. Ciarallo, Ph.D.

__________________________________________  
Xinhui Zhang, Ph.D.

__________________________________________  
Joseph F. Thomas Jr., Ph.D.  
Dean, School of Graduate Studies
ABSTRACT

Soneji, Hitesh Deepak. M.S. Egr., Department of Biomedical, Industrial, and Human Factors Engineering, Wright State University, 2006. A Comparison of Agent-Based Optimization Approaches Applied to the Weapons to Targets Assignment Planning Problem.

Real-world complex optimization problems are difficult to solve. Agent-based optimization approaches have proved useful in solving a wide variety of problems including optimization problems. Agent-based techniques can be used in military planning for solving allocation problems such as the weapons to targets assignment problem. Classical methods like linear programming (LP) have been used for solving weapons to targets assignment problems. LP approaches provide optimal solutions quickly, but in real-time planning when there are minor changes to input, LP exhibits widely varied solutions. This can be a problem in practice.

This research study considers two agent-based optimization approaches, the Stable Marriage Algorithm (SMA) and the Ant-Colony Optimization (ACO) algorithm, for solving the weapons to targets assignment problem. In real-time defense planning and re-planning scenario, the effect of the input data changes on the solutions provided by SMA and ACO is observed. An interactive tool is developed in Visual Basic 6.0 for performing the assignment of weapons to targets using either of the agent-based optimization algorithms. An empirical analysis for determining the best parameter settings for finding good solutions for ACO algorithm is carried out. The performance of SMA and ACO is compared in terms of solution quality and persistence characteristics. Results indicate better performance of SMA than ACO in terms of persistence. In terms of solution quality, ACO provides solutions with lower assignment cost values than SMA.
## Contents

1 Introduction .................................................. 1
   1.1 Objectives of Research ................................. 2
   1.2 Outline .................................................. 3

2 Background ................................................... 4
   2.1 Weapons to Targets Assignment (WTA) Problem .......... 4
   2.2 WTA Problem Formulation ............................... 8
   2.3 Stable Marriage Algorithm (SMA) ....................... 9
      2.3.1 Gale and Shapely (GS) Algorithm .................. 9
   2.4 Ant-Colony Optimization (ACO) Algorithm ............... 10

3 Weapons to Targets Assignment Tool Prototype .......... 13
   3.1 Objectives .............................................. 13
   3.2 Detailed Design of the Prototype ....................... 14
      3.2.1 Welcome/Login Screen .............................. 14
      3.2.2 Opening Screen ................................... 14
      3.2.3 Base Setup Screen ................................ 15
      3.2.4 Target Selection Screen .......................... 16
      3.2.5 Weapon Selection Screen .......................... 18
   3.3 Use of the Prototype Tool ............................... 19

4 Empirical Study ............................................... 22
4.1 Analysis of Heuristics .................................................. 22
4.2 SMA Implementation .................................................. 24
    4.2.1 Pseudo code for SMA ......................................... 24
4.3 ACO Implementation .................................................. 27
    4.3.1 Pseudo code for ACO ......................................... 30
4.4 Experimental Study .................................................. 32
    4.4.1 Parameter Setting for Determining a Best ACO ............ 32
    4.4.2 Comparing SMA vs ACO ..................................... 34
4.5 Experimental Results ................................................ 34
    4.5.1 Test Problems .................................................. 35
    4.5.2 Parameter Setting for ACO ................................... 35
    4.5.3 SMA vs ACO .................................................. 37

5 Conclusions and Summary ........................................... 43
    5.1 Conclusions ....................................................... 43
    5.2 Thesis Accomplishments .......................................... 44
    5.3 Future Research Avenues ........................................ 44

Bibliography ............................................................. 46
List of Figures

2.1 Pseudo code of SMA (Richardson, 2003) ........................................ 11

3.1 Login Screen ............................................................................. 15
3.2 Opening Screen ....................................................................... 16
3.3 Base Setup Screen .................................................................... 17
3.4 Target Selection Screen .............................................................. 18
3.5 Weapon Selection Screen ............................................................ 19
3.6 Target Selection Screen with Enabled Targets Marked on Map .... 21
## List of Tables

4.1 Assignment Cost for Best Ant Solution ........................... 36
4.2 Assignment Cost for Each Ant Solution ............................ 36
4.3 Results for 100 x 100 problem set by LP, SMA, and ACO ........ 38
4.4 Results for 200 x 200 problem set by LP, SMA, and ACO ........ 39
4.5 Results for 300 x 300 problem set by LP, SMA, and ACO ........ 40
4.6 Ratio of Changes in ACO/Changes in SMA ......................... 41
4.7 Ratio of Assignment Cost by ACO/Assignment Cost by SMA ........ 42
Acknowledgement

I wish to express my gratitude to Dr. Raymond R. Hill, Dr. Xinhui Zhang, and Dr. Frank W. Ciarallo for serving as my supervisory committee members.

My deepest gratitude goes to my advisor Dr. Raymond R. Hill, whose support and guidance was extremely important throughout the completion of this thesis. He has been a great supervisor and mentor throughout my Masters at Wright State University. His advice, expertise, patience, and cooperation has been extremely helpful.

My utmost appreciation goes to my parents, Deepak Soneji and late Varsha Soneji, source of my strength, for their eternal support and love. I would like to extend my thanks to all my family members for their continuous support and encouragement.

I would like to thank Dr. Xinhui Zhang for his constant support and help with Xpress tool. My special acknowledgement goes to my colleagues at Wright State University, especially those in the BIE department and AMOS laboratory for their support and encouragement.
Chapter 1

Introduction

Real-world combinatorial optimization problems are complex and practically difficult to solve. Agent-based optimization approaches have proved useful for finding solutions for such optimization problems, sometimes in a more flexible and efficient way.

Agent-based approaches have been considered for solving assignment problems such as the weapons to targets assignment problem in a military planning and re-planning context (Hill, et al., 2004; Lee and Lee, 2003; Lee, Lee, and Su, 2002; and Lee, Su, and Lee, 2002). In this scenario, weapons and targets are considered as individual entities or agents with decision making capabilities. The weapons to targets assignment problem is an NP-hard problem and methods like linear programming (LP), simulated annealing (SA), genetic algorithms (GA), and Ant-Colony Optimization (ACO) algorithms have been employed for solving these assignment problems (Lee, Lee, and Su, 2002).

Often assignment problems must be re-solved due to changes in input data. Classical methods such as LP exhibit potentially many changes to solutions when there are minor changes to inputs. This has been termed a “lack of persistence” in the LP solution. In a dynamic tactical environment, when a new assignment is planned, a persistent assignment model plays an important role when small changes to input data leads to drastically different output solutions (Castro, 2002).
Preliminary experiments were conducted by Hill et al. (2004) and Nadkarni (2004) to study the persistence characteristics of LP and the Stable Marriage Algorithm (SMA). Optimal solutions are quickly provided by solving the assignment problem using an LP approach. However, in real-time planning and re-planning applications, the lack of persistence demonstrated by an LP may not be preferred, particularly when the change is expensive. This early work uncovered interesting persistence qualities with the SMA. This research extends past research by adding the ACO to the comparison.

1.1 Objectives of Research

This research builds upon the prior work by Hill et al. (2004) and Nadkarni (2004) by examining another popular agent-based optimization approach, the Ant-Colony Optimization (ACO) for solving the weapons to targets assignment problem. In real-time planning, efforts for re-planning may be required when the status of the system changes because of input variations. The algorithms implemented, SMA and ACO, are useful in studying the effects of minor input variations such as adding or deleting a weapon or a target on the solutions provided by these algorithms.

An interactive tool, the Weapons to Targets Assignment Tool is developed in Visual Basic 6.0, and is used to perform an assignment of weapons to selected targets in a defense planning and re-planning scenario. The tool uses either the SMA or ACO algorithms for performing the assignment of weapons to targets. An empirical analysis is carried out for determining the best parameter settings for an ACO algorithm for finding good solutions. The purpose of this research is to compare the performance of SMA and ACO in terms of the quality of the solution and persistence of that solution.
1.2 Outline

This section provides the outline of the thesis. Chapter 2 provides a brief introduction to the weapons to targets assignment problem and discusses the optimization algorithms, SMA and ACO, along with their applications. Chapter 3 describes the interactive planning tool developed for performing the assignment of weapons to targets. Chapter 4 describes the research methodology and the implementation of SMA and ACO along with the results of an empirical analysis comparing SMA and ACO performance. Chapter 5 provides a summary, conclusions, and avenues for future research.
Chapter 2

Background

2.1 Weapons to Targets Assignment (WTA) Problem

On modern battlefields, the proper assignment of weapons to targets is considered an important task for a planner (Lee, Su, and Lee, 2002). The weapons to targets assignment (WTA) problem has the objective of assigning weapons to targets while maximizing the expected damage value exerted by own-force assets. WTA problems are difficult to solve to optimality if the number of weapons and targets are large. Computation time to find the optimal solutions increases rapidly with the size of the problem. Traditional methods such as branch and bound used to exactly solve these types of problems result in exponential increase in computational requirements (Lee and Lee, 2003). Dynamic programming, separable convex objective functions, graph theory, etc., have been used for finding solutions to WTA problems (Lee, Su, and Lee, 2002). Linear programming (LP) has been widely used for solving the WTA problem in military planning to match both conventional and nuclear weapons to targets of military importance (Hill, et al., 2004).

A target-assignment problem can be formulated into an LP form. Manne (1958) showed that despite apparent nonlinearities, it is possible to recast the problem into LP form by
fairly minor modifications of the original target-assignment problem, and a transformation of variables. This provides a close approximation to the original target-assignment problem.

Day (1966) presented a three-stage optimization method for assigning weapons to targets by means of nonlinear programming. The targeting problem was sub-divided into smaller problems with a reduced number of variables. The procedure involved optimizing the sub-divided targeting problem and estimating complex damage functions. A considerable reduction in the dimensions of the optimization problem was achieved along with the economic use of data. The objective defined was to maximize the expected damage on the target system.

A decision support tool was developed by Might (1987) to determine the most cost-effective munition and aircraft combination against a target. The objective of the linear program formulation was to maximize the number of targets destroyed, subject to aircraft, munitions, targets, weather, and budget constraints. The important benefit of Might’s methodology was accounting for budget and resource constraints in the planning process.

Wacholder (1989) presented a neural network-based algorithm for performing an optimized assignment of weapons to targets. The formulation was developed for solving a static WTA problem. The advantage of this algorithm was that fast and accurate results to difficult decision problems were obtained by implementing the algorithm in a special-purpose hardware circuit.

Green, Moore, and Borsi (1997) applied a goal programming approach for solving the WTA problem provided the prioritized goals are specified. The Arsenal Exchange Model (AEM), used to assign weapons to targets with the objective of maximizing the damage expectancy, was modified to obtain feasible integer solutions from the continuous LP solutions. An integer solution was produced by truncating the non-integer valued variables in the continuous solution. A heuristic developed improved the overall performance of AEM and yielded good solutions.
Koewler (1999) developed a prototype tool for scheduling and resource allocation to help solve combat planning problems. The computer-based tool helps combat planners task air resources to targets with the balance between ease of use, accurately defining and solving the scheduling and allocation problem, and providing good solutions within an operationally useful amount of time. The methodology developed used a combination of concepts: project scheduling, object-oriented programming, and genetic-algorithms (GA).

A substantial challenge involves determining a correct balance between weapons investment and information assets (Yost and Washburn, 2000). A new optimization methodology by Yost and Washburn (2000) combined air-to-ground attack assets and bomb-damage assessment (BDA) sensors in a single allocation model. The integrated model helps analyze the relationship between transformation (attack) assets and information (sensor) and thereby recommend actions based on assessment probabilities.

Static and dynamic allocation models were developed by Castro (2002) for assigning agents (aircrafts) to tasks (targets) with a mixed-integer program (MIP). Both models, implemented in GAMS using both the CPLEX and XA solvers, ensured that the aircraft had sufficient range, time, and weapons in order to achieve high probability of kill against each assigned target.

Ahuja et al. (2003) proposed several exact and heuristic algorithms for solving the WTA problem. They suggested linear programming, integer programming, and network flow-based lower bounding methods in order to obtain several branch-and-bound algorithms for solving the WTA problem. They also proposed a very large-scale neighborhood (VLSN) search algorithm to solve the WTA problem. The WTA problem was formulated with the objective of minimizing the total expected survival value of the targets. Weapons of different types were assigned to a target ensuring that the number of weapons used is not more than the number available.

Battle functions like dynamic command and control require efficient and fast decision
support tools for optimal assignment of resources (Rosenberger, et al., 2005). Rosenberger et al. (2005) formulated the WTA problem as a linear integer programming problem with the objective of maximizing the total benefit of assignments. The authors investigated two solution methods, one a greedy heuristic method of sequential assignment of targets to sources and a second built on a branch-and-bound framework. The branch-and-bound technique was extended for assigning multiple platforms per target. This demonstrated the utility of collaborative asset planning.

Agent-based optimization methods have also been used for solving complex optimization problems. These approaches were considered for the assignment of weapons to targets in a defense planning and re-planning scenario. Methods, such as simulated annealing (SA) and genetic algorithms (GA) have been used for solving optimization problems and these methods have provided satisfactory performance in various applications (Lee, Lee, and Su, 2002). Even though SAs have shown the ability to find the optimal solution, they cannot be used in a parallel architecture to improve their search efficiency. On the other hand, GAs provide better solutions by implementing parallel search techniques. However, if the operators are not designed carefully, GAs may have poor search performance (Lee, Su, and Lee, 2002). Lee and Lee (2003) embedded ACO into a GA to improve the local search efficiency.

Lee, Su, and Lee (2002) employed GAs for solving WTA problems with the objective of minimizing the cost of assignment. With \( N \) targets and \( N \) weapons on the battlefield, two assumptions were made for solving the problem. First, all the weapons are assigned to all targets. Second, the probability of damage for every target-weapon pair is known. The simulations run by implementing the formulation showed good performance using existing GAs.

The ACO algorithm is a popular agent-based optimization approach. ACO is a class of algorithms that mimick the behavior of real ants using artificial ants. These artificial ants are capable of exploring and exploiting artificial pheromone information and then decide
upon a path based on the pheromone concentration. ACO has been widely employed as a cooperative search algorithm for solving optimization problems (Lee and Lee, 2003). The following section discusses the mathematical formulation for solving the WTA problem.

2.2 WTA Problem Formulation

The focus of the WTA problem is to perform an effective assignment of weapons to destroy targets. In this thesis, we consider the basic WTA problem, $N$ targets and $N$ weapons on a battlefield with the assumption that all targets are assigned a weapon and all weapons are assigned to targets thus forming a balanced assignment problem. We also assume that the cost $C_{ij}$ for assigning weapon $j$ to target $i$ is known for all target-weapon pairs.

The formulation, with the objective of minimizing the total assignment cost is as follows.

\[
\text{Minimize} \sum_{i=1}^{N} \sum_{j=1}^{N} C_{ij}X_{ij} \quad (2.1)
\]

subject to

\[
\sum_{j=1}^{N} X_{ij} = 1 \quad i = 1, \ldots, N \quad (2.2)
\]

\[
\sum_{i=1}^{N} X_{ij} = 1 \quad j = 1, \ldots, N \quad (2.3)
\]

\[
X_{ij} = \begin{cases} 
1 & \text{if weapon } j \text{ is assigned to target } i \\
0 & \text{otherwise}
\end{cases} \quad (2.4)
\]

where
The application of ACO and SMA as agent-based approaches for assigning weapons to targets in the military planning context is examined in this thesis. Both SMA and ACO appear promising in assignment applications where weapons and targets preferences play an important role in calculating the cost of assignment (Hill, et al., 2004). A brief description of SMA and ACO is provided next.

2.3 Stable Marriage Algorithm (SMA)

The concept of SMA was first introduced by Gale and Shapely in 1962 for matching and admitting students to colleges (Gale and Shapely, 1962). SMA has been used in matching medical interns to hospital positions based on stated intern preferences (Hill, et al., 2004). According to McVitie and Wilson (1971), the definition of the SMA is:

“Consider two distinct sets A and B. An assignment of the members of A to the members of B is said to be a stable marriage if and only if there exist no elements a and b (belonging to A and B respectively) who are not assigned to each other but who would both prefer each other to their present partners.”

2.3.1 Gale and Shapely (GS) Algorithm

Given any set of preferences for the entities in one set towards entities in another set, the Gale and Shapely (GS) algorithm provides a stable marriage of pairings among the sets of entities. In the GS algorithm every man proposes to their most preferred woman that he has not already proposed to. Every woman proposed to considers all the received
proposals, keeps on hold the most preferred proposal and rejects all the other proposals. Each rejected man proposes to their next preferred woman on their preference list. This process of proposing continues until all men are rejected or the rejected men have finished proposing to the women in their preferred list. At the end of the process, every woman is paired to the man whom she has kept on hold. A man who has been rejected by all the women or a woman who has never received a proposal is left single (unassigned). The pseudo code for SMA by Richardson is shown in Figure 2.1.

The men and women are paired by the SMA such that there are no “unstable” pairings. An unstable pairing involves couples (a1, b1) and (a2, b2), such that man a1 prefers woman b2 over his current match b1. Similarly, man a2 prefers woman b1 over his current match b2. The SMA iteratively resolves this kind of “unstable” pairings until all pairings are stable (Hill, et al., 2004). As applied to WTA, the weapons or targets are designated to serve the role of man or woman in the assignment process.

2.4 Ant-Colony Optimization (ACO) Algorithm

Ant-colony algorithms are a relatively new, yet extremely interesting, addition to the suite of meta-heuristic approaches. The ant-colony optimization (ACO) approach uses path re-enforcement thereby mimicking the role of a pheromone trail laid by actual ants as a means for a set of artificial ants to iteratively adapt and converge to shortest path solutions.

Real ants, acting as a group, are adept at finding a shortest path from a nest (source) to a food source (destination) without using visual clues (Dorigo and Gambardella, 1997). A constant amount of substance, called pheromone, is deposited by each ant on the trail as the ants move from their nest to the food source and vice-versa. Other isolated ants, moving in a random direction sense this pheromone trail laid by the prior ants. Each ant following this pheromone trail deposits an additional amount of its own pheromone, thus reinforcing those components of the trail most used. The pheromone trail becomes a means of indirect
communication between the ants wherein the ants choose the path based on the amount of pheromone concentrations. After a short transitory period, shorter paths receive pheromone deposits more frequently as more ants tend to follow the shorter path thus leading the ants to the food source (or nest). This collective behavior is a form of autocatalytic behavior. The entire process is characterized by a positive feedback loop where the probability of choosing a path by an ant increases with the number of previous ants choosing the same path (Dorigo, Maniezzo, and Coloni, 1996).

Artificial ants used in optimization applications are endowed with non-realistic properties, differentiating them from actual ants to provide them with greater optimization problem-solving capabilities. Such properties include attributes such as memory and sight. As applied to optimization, providing computational memory and algorithmic look-ahead (sight) functions improve overall algorithm performance.
Researchers have applied the ACO paradigm to a wide range of problems (Dorigo, Bonabeau, and Theraulaz, 2000), and have examined a variety of methods to implement the re-enforcement mechanisms, control the movement of the ants, and improve the ability of the ants to obtain high quality solutions.

Dorigo, Maniezzo, and Colomn (1996) introduced the ACO approach for solving the traveling salesman problem (TSP). Dorigo and Gambardella (1997) solved symmetric and asymmetric TSPs with results better than results obtained using a GA, SA, and annealing-genetic algorithm (AG).

The ACO algorithm has been widely applied to vehicle routing problems (VRPs) and telecommunication routing. Barcos et al. (2004) used ant colony techniques for solving VRPs to minimize total transportation and handling costs. Bell and McMullen (2004) used an ACO approach to obtain experimental results within 1 percent of the known optimal solution for the VRPs used in their study. ACO algorithms have also been applied to solve VRPs with backhauls (Wade and Salhi, 2001), and time windows (Reimann, Doerner, and Hartl, 2002). Caro and Dorigo (1998) introduced AntNet, an adaptive, distributive, and mobile-agent based algorithm for packet routing in communication networks. Garlick and Barr (2002) developed an ACO for dynamic wavelength routing in WDM networks to minimize the total network blocking by considering the length and congestion information.

Lee, Lee, and Su (2002) proposed an immunity-based ACO algorithm for solving the WTA problem. The ACO technique has been used by Jain and Gupta (2005) for solving the operation sequencing problem with the objective of minimizing the total sum of change-over costs for a given mechanical part. Maniezzo and Carbonaro (2000) applied ACO for solving one of the main problems arising in mobile telecommunication, namely the frequency (or channel) assignment problem. Maniezzo and Colomn (1999), inspired by the behavior of the ant colonies, described and proposed a distributed heuristic algorithm for solving the Quadratic Assignment Problem (QAP).
Chapter 3

Weapons to Targets Assignment Tool Prototype

3.1 Objectives

The purpose of the Weapons to Targets Assignment Tool is to perform an assignment of weapons to selected targets in a defense planning and re-planning scenario. The tool is considered a prototype at this stage. The interface for the Weapons to Targets Assignment Tool is designed considering human factors engineering concepts such as identifying usability goals, task analysis, conceptual modeling, developing usability test plans, and analyzing usability assessments. This tool provides an efficient way of assigning a weapon to a target using SMA or ACO implementations. The assignment tool helps the user perform the following functions:

1. Base setup from which weapons are based;

2. Target selection (or de-selection);

3. Weapon selection (or de-selection);

4. Planning of the weapons-target strategy; and
5. Re-planning given changes to weapons or targets.

To set up a base, the user has a pre-defined list of existing bases from which to select. The properties of the selected base are displayed on the screen. After selecting a base, the user selects the set of targets and weapons for which the assignment is performed. The priority of assigning a weapon to a target is decided by the target’s and the weapon’s preference list. The preference lists are generated considering the assignment cost values, although in practice a targeting expert can also assign preference values. The user can enable/disable targets or weapons and re-plan the mission as necessary.

### 3.2 Detailed Design of the Prototype

The Weapons to Targets Assignment Tool prototype is developed in Visual Basic 6.0 using MS Access as the data source. The user can save the results of the assignment tasks in an Excel file to a pre-defined directory. The detailed design description of the prototype follows.

#### 3.2.1 Welcome/Login Screen

Figure 3.1 is the Welcome/Login screen for the assignment tool. For authentication purposes a login capability to the system is provided. Once the user provides a correct username and password, the Opening screen appears.

#### 3.2.2 Opening Screen

Figure 3.2 shows the Opening screen. This screen provides the user with a brief introduction of the tasks to perform. It allows the user to open any previously saved assignment results or perform a new assignment. If the user chooses to perform a new assignment, they...
are presented with the Base Setup screen. If the user chooses to open a previously saved assignment, they can browse and select among saved results files.

### 3.2.3 Base Setup Screen

The Base Setup screen as shown in Figure 3.3 provides the facility for adding and removing bases. The user has an option to select the type, name, and icon for the base. On the click of the “Add Base” pushbutton, the base name gets added to the Base Properties list box and the selected base icon is shown on the map. The user ensures the base type, name, and icon fields are selected before adding the base to the map. The Base Properties list box displays the name and the type of the base added. The properties of the base (country,
latitude, longitude, and altitude above the sea level) selected in the list box are shown below the Base Properties list box. The user can add multiple bases similarly. When the user selects the “Remove Base” pushbutton, the user is prompted with a confirm delete message box. If the user confirms the delete base action, the selected base is removed from the map. The user selects bases from the existing bases; users currently cannot create a new base. The user can change the status of the base by enabling/disabling it from selection.

3.2.4 Target Selection Screen

The Target Selection screen is depicted in Figure 3.4. The user can create new targets, edit or delete existing targets, place targets on the map, and enable/disable targets using
the Target Selection screen. When the user selects the type of target, a set of icons is provided. The user selects an icon to assign to the target when placed on the map. Target properties include latitude, longitude, distance band, weather state, time period, target importance, damage expectancy, and identification of the target as enemy or friendly. The user is prompted with message boxes to confirm actions such as creating, updating or deleting targets. When the user selects a target, its properties are displayed in the Target Properties section. The user can change the properties and update the information for the selected target. These features are currently disabled in the prototype version used in this thesis work. The user can enable/disable the targets for performing the assignment. The number of targets enabled is displayed on the screen.

Figure 3.3: Base Setup Screen
The Weapon Selection screen shown in Figure 3.5 allows the user to add or delete a weapon, or temporarily enable or disable a weapon. The user can select any weapon by selecting the type, category, and name of the weapon. After selecting the weapon, the weapon properties (availability of the weapon, its range, cost, time period, and speed) are displayed on the screen. On selecting the “Add Weapon” pushbutton, the weapon is displayed in the enabled weapons listbox. These features are disabled in the prototype version used in this thesis work. The user can enable/disable the weapons and plan the mission. The number of weapons enabled is displayed on the screen which should be equal to the number of enabled targets for solving a balanced assignment problem. When planning the mission, the tool
tip on the target icon displays the weapon number assigned to that target. The summary of the assignment is saved in an Excel spreadsheet.

If the user wants to logout from the system, they click the “Logout” pushbutton. The user is prompted with a message box to confirm this logout action.

### 3.3 Use of the Prototype Tool

Currently the user can perform a balanced assignment of weapons to targets using either the SMA or the ACO algorithm. Problems considered are symmetric. In the symmetric type of problems the number of enabled targets is equal to the number of enabled weapons whereas

![Figure 3.5: Weapon Selection Screen](image-url)
for the asymmetric problems there is an imbalance of enabled weapons or targets. The user is provided with a set of enabled targets and weapons. After an initial plan is developed, the user can enable or disable targets or weapons and resolve the problem. All results are saved in an Excel sheet. The saved results can be used for comparing the results of the current run with the results of the previous run. The following steps provide a detailed example walk-through of the tasks involved in the weapons assignment process.

Step 1: The user logs into the assignment tool and navigates to the screen with three main tabs of Base Setup, Target Selection, and Weapon Selection.

Step 2: On the Target Selection screen, a set of 100 enabled targets is provided in the Targets Status section. The user can enable or disable any target. For balanced assignments, the user ensures that the number of enabled targets equals the number of enabled weapons. All the targets are displayed on the map with a yellow triangle.

Step 3: On the Weapon Selection screen, a set of 100 enabled weapons is provided in the Weapons Status section and the user can similarly enable or disable any weapon. The user is allowed to plan the mission only when the number of enabled targets is equal to the number of enabled weapons.

Step 4: When the user clicks on the “Plan” button, either the SMA or ACO algorithms are executed using the set of enabled targets and weapons. The user is prompted with an input box where they can enter the name of the file to save the results of the assignment. All the enabled targets are displayed on the map colored in blue as shown in Figure 3.6. The tool tip on each target icon provides the weapon number assigned to that target.

Additional functionality to scroll the map in the horizontal and vertical direction is provided by the tool. This provides more area for placing targets when the problem size gets larger.
Figure 3.6: Target Selection Screen with Enabled Targets Marked on Map
Chapter 4

Empirical Study

4.1 Analysis of Heuristics

Good feasible solutions to optimization problems can often be obtained by the application of heuristic optimization algorithms. Heuristics are procedures for obtaining acceptable solutions to problems within some limited computation time (Zanakis and Evans, 1981). Heuristics are advantageous and desirable to use in many decision scenarios. Zanakis and Evans (1981) and Silver et al. (1980) mention several reasons for using a heuristic method. These reasons are combined and listed below.

1. A exact and reliable solution procedure or method for solving the mathematical problem is not available.

2. A exact reliable solution procedure or method is available but due to excessive computation time, the method is less attractive.

3. Heuristic methods are very simple to understand and easy to design and implement.

4. Heuristics can be used for learning purposes to gain insight to solve complex optimization problems.
5. Heuristics can provide good starting solutions that can drastically reduce the computational effort expended using exact solution methods.

Properties of a good heuristic as mentioned by Zanakis and Evans (1981) and Silver et al. (1980) are combined and summarized below.

1. Computational efforts should be realistic.

2. Solution provided by the heuristics should be accurate and of high quality.

3. The method should be robust i.e., it should obtain good solutions to a range of problem instances.

4. Heuristics should be simple facilitating user understanding of the approach.

One way to study the performance of a heuristic method developed for large complex optimization problems is through empirical analysis. This involves applying heuristic methods to a collection of problem instances. The performance measures for experiments on heuristics generally involves comparing the observed solution quality and computational time (Rardin and Uzsoy, 2001) to exact solution methods. According to Hooker (1994), an empirical science of algorithms is a viable alternative that requires rigorous experimental design and analysis along with empirically-based explanatory theories.

Coy et al. (2000) proposed a procedure for finding the best parameter settings for two new vehicle routing heuristics using a factorial experimental design. The procedure selects a subset of problems from the entire set of problems and finds a high quality parameter setting for each problem, and then determines a good parameter setting for the entire set of problems. Crary and Spera (1996) demonstrated the use of two optimal experimental designs, D-optimal and I-optimal, to generate input instances for measuring the empirical performance of algorithms applied to knapsack problems. Park and Kim (1998) used a nonlinear optimization method based on simplex design, to quickly and without much human
intervention, find appropriate values for parameters used in SA algorithm applications. Factorial experiments were constructed for determining the best parameter settings for neural network models by Robertson et al. (1998) as an application to financial modeling.

This research empirically compares the performance of two agent-based optimization algorithms, SMA and ACO, in terms of solution quality and persistence. In the following sections, the implementation of SMA and ACO is discussed.

4.2 SMA Implementation

In an agent-based environment, targets and weapons are considered individual agents. Agents have preference functions and communicate with each other using the target-centered SMA approach. In this approach every target proposes to a weapon depending on its preference function and every weapon either accepts or rejects the target proposal by considering its preference for a particular target. This process of proposing continues until every weapon is paired with a suitable target. The procedure finds a female-optimal stable marriage solution.

4.2.1 Pseudo code for SMA

The algorithm implemented in this research is the first procedure for the SMA discussed by McVitie and Wilson (1971). This procedure finds a stable marriage solution using the GS algorithm. The results of the assignment are stored in the integer array assignment. Thus assignment(i) is the target number to which i\textsuperscript{th} weapon is assigned. Definitions of the variables and arrays are as follows.

\begin{align*}
  n & \quad \text{Total number of targets and weapons (i.e., the size of the problem)} \\
  a & \quad \text{Number of enabled targets and weapons}
\end{align*}
\textit{m} \quad \text{Number of proposals made in each iteration}

\textit{count} \quad \text{Total number of proposals made before the final SMA results are obtained}

\textit{wc(i, j)} \quad \text{Integer array with choice number of the } j^{th} \text{ target to weapon } i

\textit{refuse(i)} \quad \text{Boolean array with the effect of the proposal for target } i

\textit{proposal(i)} \quad \text{Integer array that stores the weapon number to which } i^{th} \text{ target proposes}

\textit{assignment(i)} \quad \text{Integer array with the target number to which } i^{th} \text{ weapon is assigned}

\textit{objectiveValue} \quad \text{Variable storing the total assignment cost}

\textit{targetcounter(i)} \quad \text{Counter for target preference matrix}

\textit{targetchoice(i, j)} \quad \text{Targets preference matrix i.e., } j^{th} \text{ element on the preference list of the } i^{th} \text{ target}

\textit{weaponchoice(i, j)} \quad \text{Weapons preference matrix i.e., } j^{th} \text{ element on the preference list of the } i^{th} \text{ weapon}

\textit{assignmentCost(i, j)} \quad \text{Integer array that stores the cost of assigning weapon } i \text{ to target } j

The pseudo code for the implemented SMA follows.

\textbf{START}

\texttt{Comment: This part of the pseudo code rearranges the \textit{weaponchoice(i, j)} array for convenience in the assignment part when the weapons compare proposals.}

\texttt{BEGIN}

\texttt{FOR } i = \text{ each enabled target in the list}
\texttt{FOR } j = \text{ each enabled weapon in the list}
\texttt{SET } wc(i, \text{ weaponchoice}(i, j)) \text{ to } j
\texttt{SET } refuse(i) \text{ to True}
\texttt{SET } assignment(i) \text{ to } 0
\texttt{SET } targetcounter(i) \text{ to } 1
\texttt{SET } wc(i, 0) \text{ to } a + 1

\texttt{END}

\texttt{END}

\textbf{START}
Comment: In this part, every rejected target proposes to the next weapon in its preference list. Initially all the targets propose to their first choice.

PROPOSAL
BEGIN
SET \( m \) to 0
FOR \( i \) = each enabled target in the list
IF proposal of enabled target is rejected Then
Rejected target proposes the next weapon in its preference list
INCREMENT \( targetcounter(i) \) by 1
INCREMENT \( m \) by 1
SET \( refuse(i) \) to False
ELSE
SET \( proposal(i) \) to \(-1\)
END IF
END FOR

Comment: The program terminates if there are no proposals made by the targets.
IF \( m = 0 \) then goto FINISH
INCREMENT \( count \) by \( m \)

Comment: In the next part, all the weapons having a proposal decide whether to accept or reject the proposal made by the target.
FOR \( i \) = each enabled target in the list
IF current enabled target has proposed a weapon Then
Get the weapon number and assign it to \( j \)
IF \( wc(j, i) > wc(j, assignment(j)) \) Then
SET \( refuse(i) \) to True
ELSE
SET \( refuse(assignment(j)) \) to True
ASSIGN \( j^{th} \) weapon to target \( i \)
END IF
END IF
END FOR
END
GOTO PROPOSAL

SET \( objectiveValue \) to 0

Comment: This part computes the objective value i.e., the assignment cost
for the solution provided by SMA.

FOR $j =$ each enabled weapon in the list
    $\text{objectiveValue} := \text{objectiveValue} + \text{assignmentCost}(j, \text{assignment}(j))$
END FOR

FINISH

4.3 ACO Implementation

ACO is a multi-agent algorithm in which each ant, considered as a simple agent, derives a solution to the full assignment problem. The ACO algorithm implemented uses a number of ants equal to the number of targets. The algorithm implemented has the following characteristics.

1. Each ant chooses a weapon with a probability that is a function of the assignment cost and the pheromone concentration associated with the target-weapon pair.

2. Weapons already assigned by an ant are not considered for further assignments.

3. Pheromone concentration on each target-weapon arc representing a target-weapon pair is updated after all the ants generate their solutions.

For the first iteration, each ant chooses a target in sequence and randomly assigns a weapon to that target. Subsequent iterations make arc selections using pheromone level data. This process continues until all the weapons have been assigned to the targets. For $0 \leq \alpha \leq 1$, the probability that the $k^{th}$ ant assigns weapon $j$ to target $i$ defined by Maniezzo and Colorni (1999) is given by equation 4.1.

$$P_{ij}^k(t) = \begin{cases} 
\frac{\alpha \tau_{ij}(t) + (1 - \alpha)\eta_{ij}}{\sum_{r \notin \text{tabu}_k} \alpha \tau_{ir}(t) + (1 - \alpha)\eta_{ir}} & \text{if } j \notin \text{tabu}_k \\
0 & \text{otherwise}
\end{cases}$$

(4.1)

where
\( \tau_{ij} \)  Pheromone concentration associated with arc \((i, j)\),

\( \eta_{ij} \)  Quality of the arc in terms of cost-benefit i.e. inverse of the position of the target \(j\) in the preference list of weapon \(i\),

\( \alpha \)  User defined parameter governing the relative importance of the pheromone concentration with respect to the arc quality, and

\( \text{tabu}_k \)  Vector containing the tabu list for the \(k^{th}\) ant i.e., set of weapons assigned by ant \(k\) to targets \(i\) at iteration \(t\).

Once assigned, a weapon joins an ant’s tabu list to avoid getting assigned a second time. When all the targets are assigned the ant has a solution to WTA problem. Such a solution is constructed by each ant in the set of ants.

In equation 4.1, parameter \( \alpha \) allows the user to control the influence of the pheromone trail \(\tau_{ij}(t)\) with respect to the heuristic desirability \(\eta_{ij}\). The heuristic desirability factor \(\eta_{ij}\) for each target-weapon pair is the inverse of the position of the target in the weapons preference list. The target at the first position in the weapons preference list will thus have a high desirability factor. With \(\alpha = 0\), the targets with higher preference are favored. As the value of \(\alpha\) increases, the pheromone concentration becomes the key factor in calculating the assignment probability.

Evaporation of the pheromone trail is an important factor in the ACO. Too much pheromone biases arc selection causing ants to converge early to poor solutions. Diminishing the initial pheromone trail slowly allow the ants to arrive at a better solution by adding diversity to the search process. Trail levels are updated after all the ants have constructed their solutions. The trail levels are updated according to equation 4.2.

\[
\tau_{ij}(t + 1) = \rho \tau_{ij}(t) + (1 - \rho) \Delta \tau_{ij} \tag{4.2}
\]
\[ \Delta \tau_{ij} = \sum_{k=1}^{m} \Delta \tau_{ij}^k \]  

(4.3)

\[ \Delta \tau_{ij}^k(t) = \begin{cases} \frac{Q}{L_{Best}} & \text{if best ant has chosen coupling } (i, j) \\ 0 & \text{otherwise} \end{cases} \]  

(4.4)

where

- \( \rho \): Coefficient representing the trace’s persistence,
- \( m \): number of ants,
- \( \Delta \tau_{ij}^k \): Quantity of trace left by the \( k^{th} \) ant on the arc \((i, j)\) at the end of its solution construction,
- \( Q \): Best solution found by a linear program, and
- \( L_{Best} \): Objective function value obtained by the best ant solution.

The solutions with low value of \( L_k \) are identified as the best solutions with more concentration of pheromone trace on the arcs representing a target-weapon pair.

In the ACO algorithm implemented, at \( t = 0 \), all the target-weapon pair arcs are assigned a small amount of pheromone. Initially, tabu lists for all the ants are empty. For each iteration, ants probabilistically assign weapons to targets. Once a weapon is assigned to a target by an ant, the weapon is added to the tabu list of that ant. This process is repeated until each ant has assigned a weapon to every target. After a solution is generated by each ant, the objective function value in terms of assignment cost is calculated. The solution with the lowest assignment cost is stored as the best solution and the pheromone trail is updated globally by the best ant. Once completed, the tabu list for each ant is emptied and ants perform the assignments again. After the algorithm is run for a specified number of iterations, the best assignment solution is returned as the final assignment of weapons to targets.
4.3.1 Pseudo code for ACO

The pseudo code with the definitions of the variables and arrays of the ACO algorithm are described next.

\( L_k(k) \) \hspace{1cm} Integer array storing the objective function value for the solution obtained by \( k^{th} \) ant

\( Position \) \hspace{1cm} Variable that stores the position of a target in the weapons preference matrix

\( trailLevel(i,j) \) \hspace{1cm} Pheromone trail associated with weapon \( j \) and target \( i \)

\( AssignmentCost \) \hspace{1cm} Variable that stores the cost value for each target-weapon pair

\( HeuristicValue(i,j) \) \hspace{1cm} Heuristic value representing the quality of assigning weapon \( i \) to target \( j \)

\( trailQuantityUpdate(i,j) \) \hspace{1cm} Quantity of trace left by the ant for weapon \( j \) and target \( i \)

\( AssignmentCostValue(i,j) \) \hspace{1cm} Array that stores the assignment cost for each target-weapon pair

\( assignmentProbability(i,j) \) \hspace{1cm} Probability with which the ant assigns weapon \( j \) to target \( i \)

START

BEGIN
Comment: This part of the code reads the assignment costs and weapons preference matrix for each target-weapon pair from the database and stores them in the respective arrays.
FOR \( i \) = each enabled weapon in the list
    FOR \( j \) = each enabled target in the list
        AssignmentCostValue\( (i,j) \) := AssignmentCostValue\( (i,j) \)
        HeuristicValue\( (i,j) \) := 1/Position
    END FOR
END FOR
Comment: This part of the code initializes the probability of assigning a weapon to a target, the trail levels, and quantity of trail for each target-weapon pair.

FOR $i =$ each enabled target in the list
    FOR $j =$ each enabled weapon in the list
        SET $assignmentProbability(i, j)$ to 0.5
        SET $trailLevel(i, j)$ to 0.5
        SET $trailQuantityUpdate(i, j)$ to 0
    END FOR
END FOR

Comment: This part of the code sets the value of $L_k(k)$ to 0

FOR $k =$ 1 to maximum number of ants
    SET $L_k(k)$ to 0
END FOR

Comment: In the next part, for every iteration each ant generates a solution by assigning weapons to targets depending upon the trail level.

FOR $t =$ number of iterations
    IF $(t = 1)$ Then
        FOR $k =$ 1 to maximum number of ants
            Each ant randomly assigns weapons to targets and generates an initial solution
            Compute $L_k$ for each ant
        END FOR
    END IF
    For each target-weapon pair
        Find the ant with lowest value of $L_k$
        Compute $trailQuantityUpdate(i, j)$ and $trailLevel(i, j)$
        SET $trailQuantityUpdate(i, j)$ to 0
        SET $L_k(k)$ to 0
    ELSE
        FOR $k =$ 1 to maximum number of ants
            FOR $i =$ each enabled target in the list
                FOR $j =$ each enabled weapon in the list
                    IF weapon $j$ is not assigned to target $i$ Then
                        Compute $assignmentProbability(i, j)$
                    END IF
                    Randomly assign weapon $j$ to target $i$ if more than one weapon has same value of assignment probability and put the weapon $j$ in the tabu list of $k^{th}$ ant
                    SET assignment probability of weapon $j$ to all other targets to 0
                END FOR
            END FOR
        END FOR
    END IF
Compute $L_k$ for each ant.
4.4 Experimental Study

The objective of this research is to understand the functionality and application of SMA and ACO for solving assignment problems. An experimental design was defined and used to compare SMA and ACO performance.

4.4.1 Parameter Setting for Determining a Best ACO

Parameter setting for an ACO algorithm is an important task for finding good solutions. The parameters considered in the analysis are as follows.

1. Relative importance of the pheromone trail ($\alpha$): This factor controls the relative importance of the trace $\tau_{ij}(t)$ with respect to the heuristic desirability factor $\eta_{ij}$. With $\alpha = 0$, the trail level is not considered in computing the assignment probability thereby yielding a stochastic greedy algorithm. For higher values of $\alpha$ the ACO algorithm exhibits stagnation behavior without finding good solutions as too much importance is given to the pheromone trail and therefore ants choose the arcs chosen
by other ants in the past. Setting $\alpha$ to central values provides good solutions (Dorigo, Maniezzo, and Colorni, 1996), but the best value of $\alpha$ is problem dependent.

2. Pheromone trail persistence ($\rho$): This coefficient controls the accumulation of pheromone trace on the arcs. This provides the ant with the possibility of forgetting part of the previously gained experiences in order to exploit new solutions.

3. Number of ants ($m$): The number of ants is assumed to be equal to the number of targets. Research by Maniezzo and Colorni (1999) shows that the overall performance of the ACO is not influenced by the number of ants.

4. Number of iterations ($t$): The ACO algorithm needs to run for a specified number of iterations in order to obtain a best solution. The number of iterations should be set to a value such that the algorithm never falls into stagnation behavior, where all the ants perform the same assignments. This factor helps in describing the performance of the ACO algorithm.

5. Quantity of pheromone trace on each arc ($\Delta \tau_{ij}^k$): This parameter can be set to two values as shown in equations 4.5 and 4.6. Equation 4.5 updates the pheromone trail with the quantity computed by the best ant sequence found at the end of each iteration.

$$\Delta \tau_{ij}^k(t) = \begin{cases} \frac{Q}{L_{best}} & \text{if best ant solution has chosen coupling (i, j)} \\ 0 & \text{otherwise} \end{cases}$$ (4.5)

Equation 4.6 performs the pheromone update by considering each ant’s solution.

$$\Delta \tau_{ij}^k(t) = \begin{cases} \frac{Q}{L_k} & \text{if } k^{th} \text{ ant has chosen coupling (i, j)} \\ 0 & \text{otherwise} \end{cases}$$ (4.6)

The approach would be to first identify a good setting of the above discussed parameters and then run simulations to perform a complete analysis of the ACO model to find best values of the parameters.
### 4.4.2 Comparing SMA vs ACO

In this section we describe ways to compare the performance of SMA and ACO. We evaluate and compare the characteristics of the implemented algorithms based on the following performance measures.

1. Solution quality in terms of assignment cost: Evaluating the quality of the solution is a real challenge. Both the algorithms are run with the objective of minimizing the assignment cost. The assignments performed by SMA and ACO are evaluated by comparing the value of assignment cost. The algorithm with the lower value of assignment cost for the same set of problems is considered a better solution.

2. Persistence: Optimization formulations are modified by persistence techniques in order to minimize the changes to the previous solution when the input conditions change (Hill, et al., 2004). We examine SMA and ACO for persistence characteristics without modifying the structure of the optimization formulation. When the number of changes in the new solution obtained by resolving the problem is equal to the number of changes made in the problem, then the solution is a maximally persistent solution. The objective is to minimize the deviations of the new solution from the old solution values without specifically adding problem structure to force the persistence.

We examine and compare the performance of SMA and ACO solutions under symmetric preference structures i.e., when the number of targets is equal to the number of weapons.

### 4.5 Experimental Results

SMA and ACO algorithms were tested on an Intel Pentium 4 - CPU with 2.8 GHz processor and 512 MB RAM system.
4.5.1 Test Problems

Standard assignment test problems from Beasley (2006) were employed for 100, 200, and 300 entities. These three assignment test problems were considered as weapons to targets assignment problems with values considered as the assignment cost for each target-weapon pair. The cost of each target-weapon pair is used for generating weapons and targets preference matrices for SMA and ACO.

4.5.2 Parameter Setting for ACO

The problem set with 100 entities, treated as targets and weapons, was used as base problem for determining the best parameter settings for ACO. Research by Dorigo, Maniezzo, and Colorni (1996); and Maniezzo and Colorni (1999) showed that good performance was achieved with a higher number of ants $m$. In the ACO implementation, we set the number of ants $m$ always equal to the number of targets $n$. We also tested the stagnation behavior of ACO algorithm, where all the ants construct the same solution after a number of cycles. This indicates that no more solutions can be explored by the ants and the current best solution cannot be improved further. It was observed that after 4 iterations, all the ants produced the same solution and this solution was reported as the best solution by ACO. Hence, we set the number of iterations ($t$) to 4. This shows that the algorithm finds good solutions quickly.

Experimental study with various values of $\alpha$ and $\rho$ was carried out and the behavior of solution quality of ACO was studied. Values of $\alpha$ and $\rho$ were changed and the solution quality by the pheromone trace led by the best ant and each ant were determined. The pheromone trail is updated globally after each ant has generated its solution.

Table 4.1 shows the results obtained by the best ant solution for various values of $\alpha$ and $\rho$. Each run was performed for 4 iterations.
Table 4.1: Assignment Cost for Best Ant Solution

<table>
<thead>
<tr>
<th>Alpha (α)</th>
<th>Trail Persistence (ρ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3</td>
<td>0.5</td>
</tr>
<tr>
<td>0.4</td>
<td>473</td>
</tr>
<tr>
<td>0.5</td>
<td>459</td>
</tr>
<tr>
<td>0.6</td>
<td>458</td>
</tr>
<tr>
<td>0.7</td>
<td>461</td>
</tr>
</tbody>
</table>

Table 4.2: Assignment Cost for Each Ant Solution

<table>
<thead>
<tr>
<th>Alpha (α)</th>
<th>Trail Persistence (ρ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3</td>
<td>0.5</td>
</tr>
<tr>
<td>0.4</td>
<td>620</td>
</tr>
<tr>
<td>0.5</td>
<td>602</td>
</tr>
<tr>
<td>0.6</td>
<td>584</td>
</tr>
<tr>
<td>0.7</td>
<td>665</td>
</tr>
</tbody>
</table>

Good solutions were found for $0.8 \leq ρ \leq 0.95$ and the best result was obtained with $ρ = 0.85$. Central values of $α$ were found to provide good solutions with the best obtained with $α = 0.6$. Also, we found that the quantity of pheromone trace updated by the best ant solution provided better results in terms of solution quality than the update performed by each ant solution. We set the parameters to the experimentally determined best values of $α = 0.6$, $ρ = 0.85$, $t = 4$, $m =$ number of targets, and pheromone update performed by the best ant solution. With the ACO parameters set to the best values, the algorithm found
best solutions for all the three problem sets.

4.5.3 SMA vs ACO

The performance of SMA and ACO was compared on the basis of persistence and solution quality in terms of assignment cost. In this section we discuss the results obtained by running the SMA and ACO algorithms for the 100, 200, and 300 entities problem sets.

In a real-time planning scenario, when the input conditions change the optimization formulations can be modified so that changes to the previous solutions are minimized. In scenarios where multiple targets or weapons needs to be changed or dropped, it is often advantageous to re-solve the problem. The new solution obtained should ideally be persistent i.e., it should not make drastic changes to the previous solution. The solution persistence is examined in this thesis by temporarily disabling randomly selected targets and weapons, and then resolving the problem.

We randomly disable 1, 3, 5, and 10 targets and weapons and solve the modified problems to examine the persistence and assignment cost of the new solutions obtained. We ran 10 replications at each level of modification, each time disabling a different set of targets and weapons. The assignment cost of a solution is the objective function value of the solution.

Table 4.3 summarizes results for the 100 x 100 assignment problem with 1, 3, 5, and 10 targets and weapons disabled randomly. In each case, 10 replications with 4 iterations were run. As compared to ACO, SMA was found to be significantly more persistent at each entity modification level. The number of changes in the SMA solutions is significantly less than number of changes in the ACO solution showing SMA to be more persistent. The average number of changes in the new solution increases with the number of entities modified, both for SMA and ACO. On an average, solutions by SMA show that there are 4.6, 14.5, 17.1, and 31.9 changes to the original assignments at the 1, 3, 5, and 10 level of modification,
respectively. This is significantly lower than the corresponding average changes by ACO solutions and higher than the average changes observed by Nadkarni (2004). However, we use both the targets preference and weapons preference matrices for our SMA approach. Each target proposes to the weapons in its preference matrix. Each proposed weapon accepts or rejects the proposal by the target depending upon its preference function. This approach of considering both the target and weapon choices may make SMA more sensitive to entity modifications as compared to SMA implemented by Nadkarni (2004). This is a conjecture and further investigation and research is required to prove or change the conjecture.

Table 4.3: Results for 100 x 100 problem set by LP, SMA, and ACO

<table>
<thead>
<tr>
<th>Number of Targets and Weapons Modified</th>
<th>LP数</th>
<th>SMA数</th>
<th>ACO数</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average Assignment Cost</td>
<td>Average Assignment Cost</td>
<td>Average Number of Changes</td>
</tr>
<tr>
<td>0</td>
<td>305</td>
<td>522</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>302.5</td>
<td>514.5</td>
<td>4.6</td>
</tr>
<tr>
<td>3</td>
<td>299.5</td>
<td>504</td>
<td>14.5</td>
</tr>
<tr>
<td>5</td>
<td>297.3</td>
<td>476.1</td>
<td>17.1</td>
</tr>
<tr>
<td>10</td>
<td>288.4</td>
<td>487.2</td>
<td>31.9</td>
</tr>
</tbody>
</table>

Optimal solutions for each problem are obtained by LP. The assignment cost by LP is the lowest possible. As also found by Nadkarni (2004), LP solutions are not very persistent. Comparing the solution quality of SMA and ACO, ACO provides better solutions than SMA at each level of modification for all the problem sets. There is a significant decrease in the average assignment cost provided by ACO solution at the 1, 3, 5, and 10 level of modification.

Table 4.4 and Table 4.5 summarize the results for 200 x 200 and 300 x 300 assignment
problems respectively.

Table 4.4: Results for 200 x 200 problem set by LP, SMA, and ACO

<table>
<thead>
<tr>
<th>Number of Targets and Weapons Modified</th>
<th>LP Average Assignment Cost</th>
<th>SMA Average Assignment Cost</th>
<th>ACO Average Assignment Cost</th>
<th>ACO Average Number of Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>475</td>
<td>792</td>
<td>648</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>473.2</td>
<td>794.1</td>
<td>669.2</td>
<td>94.9</td>
</tr>
<tr>
<td>3</td>
<td>470.7</td>
<td>791.3</td>
<td>667</td>
<td>101.8</td>
</tr>
<tr>
<td>5</td>
<td>467.5</td>
<td>781.5</td>
<td>664.4</td>
<td>102.1</td>
</tr>
<tr>
<td>10</td>
<td>459.4</td>
<td>760.8</td>
<td>644.9</td>
<td>107</td>
</tr>
</tbody>
</table>

ACO provides better solutions for 200 x 200 and 300 x 300 problem sets in terms of solution quality. The value of the assignment cost is quite low as compared to SMA at each modification level. On the other hand, the number of changes by SMA solution is significantly less than the changes by ACO solution for the same modified problems.

Table 4.6 provides the ratio of the number of changes in an ACO solution to the number of changes in an SMA solution. This measure decreases with the increase in the number of modifications, for each problem set. At certain modification levels, ACO could be more persistent and thus preferred over SMA. However, this is a just a conjecture based on available data and further investigation is required to prove the conjecture.

For every problem set, we find that the SMA is more persistent than ACO at each modification level. However, ACO provides better solutions in terms of assignment cost as compared to SMA. SMA could be a preferred approach despite the higher assignment cost values in scenarios where more persistent solutions can be easily implemented.

Table 4.7 provides the ratio of the assignment cost by ACO solution to the assignment...
Table 4.5: Results for 300 x 300 problem set by LP, SMA, and ACO

<table>
<thead>
<tr>
<th>Number of Targets and Weapons Modified</th>
<th>LP</th>
<th>SMA</th>
<th>ACO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Average Assignment Cost</td>
<td>Average Assignment Cost</td>
<td>Average Number of Changes</td>
</tr>
<tr>
<td>0</td>
<td>626</td>
<td>1055</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>624</td>
<td>1042.2</td>
<td>7.3</td>
</tr>
<tr>
<td>3</td>
<td>620.8</td>
<td>1030.9</td>
<td>17.1</td>
</tr>
<tr>
<td>5</td>
<td>616.9</td>
<td>981.8</td>
<td>26.2</td>
</tr>
<tr>
<td>10</td>
<td>609.5</td>
<td>971</td>
<td>45.6</td>
</tr>
</tbody>
</table>

cost by SMA solution. This measure shows fairly constant difference between the objective function values by ACO and SMA for each problem set at each modification level.
Table 4.6: Ratio of Changes in ACO/Changes in SMA

<table>
<thead>
<tr>
<th>Number of Targets and Weapons Modified</th>
<th>100 x 100</th>
<th>200 x 200</th>
<th>300 x 300</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>7.28</td>
<td>17.57</td>
<td>19.10</td>
</tr>
<tr>
<td>3</td>
<td>2.74</td>
<td>7.49</td>
<td>8.20</td>
</tr>
<tr>
<td>5</td>
<td>2.63</td>
<td>4.46</td>
<td>5.66</td>
</tr>
<tr>
<td>10</td>
<td>1.50</td>
<td>2.79</td>
<td>3.50</td>
</tr>
<tr>
<td>Number of Targets and Weapons Modified</td>
<td>100 x 100</td>
<td>200 x 200</td>
<td>300 x 300</td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-----------</td>
<td>-----------</td>
<td>-----------</td>
</tr>
<tr>
<td>1</td>
<td>0.84</td>
<td>0.84</td>
<td>0.84</td>
</tr>
<tr>
<td>3</td>
<td>0.86</td>
<td>0.84</td>
<td>0.84</td>
</tr>
<tr>
<td>5</td>
<td>0.90</td>
<td>0.85</td>
<td>0.88</td>
</tr>
<tr>
<td>10</td>
<td>0.83</td>
<td>0.85</td>
<td>0.87</td>
</tr>
</tbody>
</table>
Chapter 5

Conclusions and Summary

5.1 Conclusions

In this thesis, we examined the ACO and SMA approaches for solving the weapons to targets assignment problem. Results verify that SMA provides more persistent results as compared to ACO at each entity modification level. While solving assignment problems, when the input conditions change, persistent solutions minimize the changes to existing solutions. To apply ACO for solving assignment problems, we experimentally studied the functioning of ACO. We ran a study and determined the best parameter settings for ACO to find good solutions. Comparing ACO and SMA in terms of solution quality proved ACO the preferred approach for solving balanced assignment problems. ACO provided higher quality solutions than SMA but these solutions are not very persistent. In real-time planning scenarios, when the input conditions change, the new solution should be persistent so that it can be easily implemented. A more persistent solution could be preferred over a low assignment cost solution. In such cases, SMA could be a preferred approach than ACO.
5.2 Thesis Accomplishments

This research extends prior work by Hill et al. (2004) and Nadkarni (2004) for solving the weapons to targets assignment problem by adding another agent-based algorithm, ACO. We examined the performance of ACO for solving balanced assignment problems. First, we found the best parameter settings for ACO and then ran SMA and ACO algorithms for solving problem sets by Beasley (2006). Comparing the performance of SMA and ACO, the results showed improved persistence among solutions generated by SMA. However, in terms of solution quality, ACO provided better solutions than SMA.

5.3 Future Research Avenues

Further research in considering real-time problems should be implemented. This thesis research considers standard problem sets by Beasley (2006). The preferences of weapons to targets and targets to weapons play an important role in the cost or value of the assignment. We generated the preference matrices for weapons and targets using the assignment cost matrix. The decision regarding the optimal assignment of weapons to targets depends on various factors like cost of the weapon, target importance, damage expectancy, target and weapon type, distance band, weather, time period, etc. This detailed information can be captured to extend the functionality of the software tool to facilitate the elicitation of weapons to targets preferences. Such real weapons to targets assignment problems, with large number of weapons and targets, can be used to perform the assignment and gain deeper insights into the performance of SMA and ACO. This approach might be useful in a dynamic scenario for performing the assignment.

The number of targets and weapons might not be the same in real-time planning. Scenarios with asymmetric preference matrices have not been considered in this study. This research work accounts for symmetric matrices for performing balanced assignment
of weapons to targets. This could be further extended for unbalanced assignments using asymmetric matrices.

ACO formulation considering both, weapons and targets preference matrices, should be designed and the performance of ACO should be verified. Higher dimension problem sets should also be run to determine the efficiency of SMA and ACO. Realistic scenarios can be developed and implemented using this assignment tool. The present tool can be further extended to incorporate map controls like adding/deleting a base, adding/deleting targets and weapons, and zoom in and zoom out.

Algorithm execution time is not considered in this study. The SMA and ACO may have different execution time for solving similar problems. A comparative study of SMA and ACO in terms of time taken to solve larger problems could be carried out. Finally, an efficient programming language should be implemented for solving SMA and ACO. (Visual Basic 6.0 is nice for interfaces but not computationally efficient).


International Command and Control Research and Technology Symposium, The Future of C2, McLean, VA.


