INTELLIGENT SEARCH AND ALGORITHMS FOR OPTIMAL ASSIGNMENT OF AIR FORCE RESOURCES IN OPERATIONS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

EMRE RIZVANOĞLU

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
ELECTRICAL AND ELECTRONICS ENGINEERING

NOVEMBER 2008

Approval of the thesis:

INTELLIGENT SEARCH AND ALGORITHMS FOR OPTIMAL ASSIGNMENT OF AIR FORCE RESOURCES IN OPERATIONS

submitted by **EMRE RIZVANOĞLU** in partial fulfillment of the requirements for the degree of **Master of Science in Electrical and Electronics Engineering Department, Middle East Technical University** by,

Prof. Dr. Canan Özgen ___________
Dean, Graduate School of **Natural and Applied Sciences**

Prof. Dr. İsmet Erkmen ___________
Head of Department, **Electrical and Electronics Engineering**

Prof. Dr. Kemal Leblebicioğlu ___________
Supervisor, **Electrical and Electronics Engineering Dept., METU**

Examining Committee Members:

Prof. Dr. Mübeccel Demirekler _____________
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Kemal Leblebicioğlu _____________
Electrical and Electronics Engineering Dept., METU

Prof. Dr. Yasemin Yardımcı Çetin _____________
Informatics Institute, METU

Assist. Prof. Dr. Afşar Saranlı _____________
Electrical and Electronics Engineering Dept., METU

Assoc. Prof. Dr. S. Gökhun Tanyer _____________
Institute Vice President, TÜBİTAK UEKAE

Date: ______________

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name: Emre Rızvanoğlu

Signature :

ABSTRACT

INTELLIGENT SEARCH AND ALGORITHMS FOR
OPTIMAL ASSIGNMENT OF AIR FORCE RESOURCES IN OPERATIONS

Rızvanoğlu, Emre
M.S., Department of Electrical and Electronics Engineering
Supervisor: Prof. Dr. Kemal Leblebicioğlu

November 2008, 115 Pages

The growing extent and variety of present military operations forces to use the resources in hand at its best. Especially, the optimum usage and assignment of limited number of the air force resources to missions will provide a considerable advantage in the battle field. The problem of finding the feasible and optimum assignment has been known to be studied; yet performing the process faster is still a topic that captures researchers' attention because of the computational complexity that the assignment problem involves within.

In this thesis, exploring the optimal assignment of fleets/aircrafts to targets/groups of targets is going to be performed via algorithms and heuristics. As the best choice for finding the exact solution, Branch-and-Bound algorithm, which is an intelligent way of searching for the solution on a solution tree where the nodes with potential of not leading to the solution are fathomed, has been investigated and applied according to the specific problem needs. The number of nodes on the search tree increases exponentially as the problem size increases. Moreover; as the size of the assignment problem increases, attaining the solution solely by Branch-and-Bound algorithm is definitely computationally expensive due to memory and time requirements. Therefore, Genetic algorithm which can provide good solutions in a relatively short time without having computational difficulties is considered as the second algorithm. Branch-and-Bound algorithm and Genetic algorithm are separately used for obtaining the solution. Hybrid algorithms which are combinations of Branch-and-Bound and Genetic algorithms are used with heuristics for improving the results.

Keywords: Assignment Problem, Branch-and-Bound Algorithm, Genetic Algorithm

ÖZ

OPERASYONLARDA HAVA KUVVETLERİ KAYNAKLARININ EN İYİ ATANMASI İÇİN AKILLI ARAMA VE ALGORİTMALAR

Rızvanoğlu, Emre
Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü
Tez Yöneticisi: Prof. Dr. Kemal Leblebicioğlu

Kasım 2008, 115 Sayfa

Askeri operasyonların çeşitliliği ve büyüyen kapsamları, bu operasyonlarda kaynakların en iyi şekilde kullanılmalarını zorunlu hale getirmektedir. Özellikle sınırlı sayıdaki hava konuşlu kaynakların en iyi kullanılması ve görevlere atanması harp alanında önemli bir yarar sağlayacaktır. En iyi ve olanaklı atamaların bulunması problemi bilinen ve üzerine uzun süredir çalışılan bir sorundur. Öte yandan, bu çözüm sürecinin daha hızlı şekilde gerçekleştirilmesi problemin kökeninde barındırdığı işlemsel karmaşıklıktan dolayı hala araştırmacıların ilgisini çeken bir konudur.

Bu tez kapsamında, filoların(uçakların) hedeflere(hedef gruplarına) en iyi atamasının araştırılması, algoritmalar ve buluşsallar kullanılarak araştırılacaktır. Kesin sonucu bulmak için kullanılabilecek en iyi algoritma seçimi olan, Böl&Sınırla algoritması araştırılıp bu belirli problemin gereksinimleri doğrultusunda uygulanmaktadır. Bu algoritma, çözümü bir çözüm ağacı üzerinde akıllı bir şekilde aramak ve çözümü barındırmaması muhtemel bölümlerin budanması esasına dayanmaktadır. Öte yandan, bahsi geçen atama probleminin boyutu büyüdüğünde, çözüm ağacının çok büyüyüp hafıza ve zaman kısıtları yaratması sebebiyle, Böl&Sınırla algoritması tek başına yeterli kalmamaktadır. Çünkü, çözüm ağacının büyüklüğü, atama probleminin büyüklüğü arttıkça üssel şekilde artmaktadır. Bu sebeple, daha kısa sürede ve işlemsel kısıtları daha az olan Genetik algoritma ikinci algoritma olarak seçilmiştir. Her iki algoritma ayrı ayrı uygulanacaktır. Ayrıca, bu iki algoritmanın birleşimi farklı buluşsallar ile desteklenerek, elde edilen sonuçları daha iyi hale getirmek amaçlanmaktadır.

Anahtar Kelimeler: Atama Problemi, Böl&Sınırla Algoritması, Genetik Algoritma

To My Family

ACKNOWLEDGEMENT

I would like to express my sincere appreciation to my supervisor Prof. Dr. Kemal Leblebicioğlu for his guidance, patience, criticism, support and encouragement throughout the study.

The idea, some techniques and definitions of this work have been sparked by the research of Tuğrul Gürdal whose help plays great role in this study. His support and presence are most appreciated.

I would also like to acknowledge my colleagues and supervisors from TÜBİTAK UEKAE – ILTAREN for their support and patience. I would especially like to thank Mehmet Alper Şahin for his help.

I am also grateful to Cihan Onmuş and Ahmet Mert, for their friendship and for being there through the most difficult times of the thesis.

Finally, my deepest thanks go to my family; my mother, my father and my brother. I would like to take this as an opportunity to thank them for their endless love, faith in me, patience and support not only for this thesis period, but also throughout my whole life.

TABLE OF CONTENTS

ABSTRACT IV

ÖZ VI

ACKNOWLEDGEMENT IX

TABLE OF CONTENTS X

LIST OF FIGURES XIII

LIST OF TABLES XIV

CHAPTERS

1. INTRODUCTION 1

1.1 PROBLEM DEFINITION 1
1.2 ASSIGNMENT PROBLEMS AND APPLICATIONS 4
1.3 AIM OF THE THESIS 5
1.4 OUTLINE OF THE DISSERTATION 7

2. ON THE CASE STUDY 8

2.1 CHAINS 10
2.2 FLEETS 15
2.3 ON THE EXTENT OF THE STUDY 16
2.4 CLASSIFICATION OF THE PROBLEM 17

3. PROBLEM FORMULATION 18

3.1 ASSIGNMENT PROBLEMS AND FORMULATIONS 18
3.1.1 Quadratic Assignment Problem 18
3.1.2 Linear Assignment Problem 19
3.1.3 Generalized Assignment Problem 20
3.2 PROBLEM FORMULATION 21

4. SOLUTION METHODS 24

4.1 SOLUTION METHODS IN THE LITERATURE 24

4.1.1 GRASP (Greedy Random Adaptive Search Procedure) ... 25
4.1.2 Ant System ... 25
4.1.3 Tabu Search ... 25
4.1.4 Simulated Annealing ... 25
4.1.5 Branch-and-Bound Algorithm ... 25
4.1.6 Genetic Algorithm ... 26
4.2 SELECTED METHODS ... 26
4.3 BRANCH-AND-BOUND ALGORITHM ... 27
4.3.1 On Branch-and-Bound Algorithm ... 27
4.3.2 Branch-and-Bound Algorithm for Assignment Problems ... 30
4.4 GENETIC ALGORITHM ... 30
4.4.1 On Genetic Algorithm ... 31
4.4.2 Genetic Algorithm for Assignment Problems ... 34

5. IMPLEMENTATION AND HEURISTICS ... 36

5.1 BRANCH-AND-BOUND ALGORITHM ... 37
5.1.1 Branch-and-Bound – Global Search (BB-GS) ... 38
5.1.1.1 Breadth-First Strategy (BreFS) ... 40
5.1.1.2 Depth-First Strategy (DFS) ... 42
5.1.1.3 Best-First Search (BFS) ... 43
5.1.2 Branch-and-Bound – Bounding Case (BB-BC) ... 45
5.1.2.1 Bound Calculation & Bounding Function Analysis ... 47
5.1.2.2 Breadth-First Strategy ... 53
5.1.2.3 Depth-First Strategy ... 54
5.1.2.4 Best-First Strategy ... 56
5.2 GENETIC ALGORITHM (GA) ... 56
5.2.1 Determination of algorithm parameters ... 60
5.2.2 Creation of Initial Population ... 60
5.2.3 Reproduction ... 60
5.2.4 Crossover ... 60
5.2.5 Crossover Feasibility Operator (CFO) ... 60
5.2.6 Mutation ... 61
5.2.7 Mutation Feasibility Operator (MFO) ... 62
5.2.8 Elitist Selection ... 62
5.3 BB-BC WITH INCUMBENT INFORMATION BY GENETIC ALGORITHM (HYBRID I) ... 62
5.4 BRANCH-AND-BOUND INITIATED BY GA PARTIAL ASSIGNMENT (HYBRID II) ... 63
5.5 NEIGHBORHOOD BRANCH-AND-BOUND INITIATED BY GA (HYBRID III) ... 65
5.5.1 Neighborhood (Distance) ... 65
5.5.2 Application in the Algorithms ... 68

6. SCENARIOS AND RESULTS ... 70

6.1 SCENARIO 1: 22 CHAINS – 8 FLEETS 72
6.1.1 BB-GS *73*
6.1.2 BB-BC *74*
6.1.3 Genetic Algorithm *74*
6.1.4 Hybrid I *77*
6.1.5 Hybrid II *78*
6.1.6 Hybrid III *79*
6.1.7 On Results of Scenario 1 ..... *81*
6.2 SCENARIO 2: 22 CHAINS – 8 FLEETS 81
6.2.1 BB-GS *81*
6.2.2 BB-BC *82*
6.2.3 Genetic Algorithm *83*
6.2.4 Hybrid I *85*
6.2.5 Hybrid II *85*
6.2.6 Hybrid III *87*
6.2.7 On Results of Scenario 2 ..... *88*
6.3 SCENARIO 3: 9 CHAINS – 8 FLEETS 89
6.3.1 BB-GS *89*
6.3.2 BB-BC *89*
6.3.3 Genetic Algorithm *90*
6.3.4 Hybrid I *92*
6.3.5 Hybrid II *93*
6.3.6 Hybrid III *94*
6.3.7 On Results of Scenario 3 ..... *96*
6.4 SCENARIO 4: 32 CHAINS – 8 FLEETS 96
6.4.1 BB-GS *96*
6.4.2 BB-BC *97*
6.4.3 Genetic Algorithm *97*
6.4.4 Hybrid I *99*
6.4.5 Hybrid II *100*
6.4.6 Hybrid III *101*
6.4.7 On Results of Scenario 4 ..... *102*

7. CONCLUSION AND COMMENTS 104

7.1 ON THE STUDY AND RESULTS 104
7.2 OUTCOME OF THE STUDY 109
7.3 FUTURE WORK 110

REFERENCES 111

LIST OF FIGURES

FIGURES

FIGURE 2.1 - SAMPLE CHAIN 9
FIGURE 2.2 - EQUIRECTANGULAR PROJECTION FOR LOCATION MAPPING 12
FIGURE 4.1 - FLOW DIAGRAM OF A GENERIC GENETIC ALGORITHM 32
FIGURE 4.2 - SINGLE POINT CROSSOVER 34
FIGURE 4.3 - TWO POINT CROSSOVER 34
FIGURE 5.1 - BRANCH-AND-BOUND – GLOBAL SEARCH: SAMPLE SEARCH TREE 39
FIGURE 5.2 - FLOW DIAGRAM OF BB-GS: BREADTH-FIRST SEARCH STRATEGY 41
FIGURE 5.3 - FLOW DIAGRAM OF BB-GS: DEPTH-FIRST SEARCH STRATEGY 42
FIGURE 5.4 - FLOW DIAGRAM OF BB-GS: BEST-FIRST SEARCH 44
FIGURE 5.5 - BRANCH-AND-BOUND – BOUNDING CASE: SAMPLE SEARCH TREE 46
FIGURE 5.6 - FLOW DIAGRAM OF BB-BC: BREADTH-FIRST SEARCH STRATEGY 53
FIGURE 5.7 - FLOW DIAGRAM OF BB-BC: DEPTH-FIRST SEARCH STRATEGY 55
FIGURE 5.8 - FLOW DIAGRAM OF THE IMPLEMENTED GENETIC ALGORITHM 59
FIGURE 5.9 - SAMPLE MUTATION ON A SAMPLE CHROMOSOME 61
FIGURE 5.10 - RETRIEVING OF PARTIAL ASSIGNMENT & ROOT NODE TO BRANCH-AND-BOUND 64
FIGURE 5.11 - ILLUSTRATION OF NEIGHBORHOOD ON TREE (DISTANCE CONCEPT) 67
FIGURE 5.12 - FLOW DIAGRAM OF THE HYBRID II ALGORITHM 68
FIGURE 6.1 - RESULTS OF SCENARIO 1 FOR GENETIC ALGORITHM (20 RUNS) 75
FIGURE 6.2 - MAXIMUM, MINIMUM AND MEAN RESULTS OF GA FOR SCENARIO 1 76
FIGURE 6.3 - RESULTS OF SCENARIO 2 FOR GENETIC ALGORITHM (20 RUNS) 83
FIGURE 6.4 - MAXIMUM, MINIMUM AND MEAN RESULTS OF GA FOR SCENARIO 2 84
FIGURE 6.5 - RESULTS OF SCENARIO 3 FOR GENETIC ALGORITHM (20 RUNS) 91
FIGURE 6.6 - MAXIMUM, MINIMUM AND MEAN RESULTS OF GA FOR SCENARIO 3 92
FIGURE 6.7 - RESULTS OF SCENARIO 4 FOR GENETIC ALGORITHM (20 RUNS) 98
FIGURE 6.8 - MAXIMUM, MINIMUM AND MEAN RESULTS OF GA FOR SCENARIO 4 99

LIST OF TABLES

TABLES

TABLE 1-1 A SIMPLE SCENARIO FOR REPRESENTING THE PROBLEM 3
TABLE 2-1 REPRESENTATION OF TARGETS 10
TABLE 2-2 TARGET TYPES AND VALUES 12
TABLE 2-3 REPRESENTATION OF CHAINS 13
TABLE 2-4 SAMPLE TARGET LIST 1 13
TABLE 2-5 SAMPLE TARGET LIST 2 13
TABLE 2-6 SAMPLE TARGET LIST & WEIGHTS 1 14
TABLE 2-7 SAMPLE TARGET LIST & WEIGHTS 2 14
TABLE 2-8 EXAMPLE OF A CHAIN VALUE CALCULATION 15
TABLE 2-9 REPRESENTATION OF FLEETS 15
TABLE 5-1 NODE STRUCTURE OF THE TREE 38
TABLE 5-2 COMPARISON OF BOUND CALCULATION METHODS 52
TABLE 5-3 GENE STRUCTURE OF GENETIC ALGORITHM 56
TABLE 5-4 CHROMOSOME STRUCTURE OF GENETIC ALGORITHM 57
TABLE 6-1 RESULTS OF SCENARIO 1 FOR BB-GS ALGORITHM 73
TABLE 6-2 RESULTS OF SCENARIO 1 FOR BB-BC ALGORITHM 74
TABLE 6-3 RESULTS OF SCENARIO 1 FOR HYBRID I ALGORITHM 77
TABLE 6-4 RESULTS OF SCENARIO 1 FOR HYBRID II - BBGS ALGORITHM 78
TABLE 6-5 RESULTS OF SCENARIO 1 FOR HYBRID II - BBBC ALGORITHM 79
TABLE 6-6 RESULTS OF SCENARIO 1 FOR HYBRID III - BBGS ALGORITHM 80
TABLE 6-7 RESULTS OF SCENARIO 1 FOR HYBRID III - BBBC ALGORITHM 80
TABLE 6-8 RESULTS OF SCENARIO 2 FOR BB-GS ALGORITHM 82
TABLE 6-9 RESULTS OF SCENARIO 2 FOR BB-BC ALGORITHM 82
TABLE 6-10 RESULTS OF SCENARIO 2 FOR HYBRID I ALGORITHM 85
TABLE 6-11 RESULTS OF SCENARIO 2 FOR HYBRID II - BBGS ALGORITHM 86
TABLE 6-12 RESULTS OF SCENARIO 2 FOR HYBRID II - BBBC ALGORITHM 86

TABLE 6-13 RESULTS OF SCENARIO 2 FOR HYBRID III - BBGS ALGORITHM 87
TABLE 6-14 RESULTS OF SCENARIO 2 FOR HYBRID III - BBBC ALGORITHM....................................... 88
TABLE 6-15 RESULTS OF SCENARIO 3 FOR BB-GS ALGORITHM 89
TABLE 6-16 RESULTS OF SCENARIO 3 FOR BB-BC ALGORITHM.. 90
TABLE 6-17 RESULTS OF SCENARIO 3 FOR HYBRID I ALGORITHM 93
TABLE 6-18 RESULTS OF SCENARIO 3 FOR HYBRID II - BBGS ALGORITHM 94
TABLE 6-19 RESULTS OF SCENARIO 3 FOR HYBRID II - BBBC ALGORITHM 94
TABLE 6-20 RESULTS OF SCENARIO 3 FOR HYBRID III - BBGS ALGORITHM 95
TABLE 6-21 RESULTS OF SCENARIO 3 FOR HYBRID III - BBBC ALGORITHM....................................... 95
TABLE 6-22 RESULTS OF SCENARIO 4 FOR BB-GS ALGORITHM 96
TABLE 6-23 RESULTS OF SCENARIO 3 FOR BB-BC ALGORITHM.. 97
TABLE 6-24 RESULTS OF SCENARIO 4 FOR HYBRID I ALGORITHM 100
TABLE 6-25 RESULTS OF SCENARIO 4 FOR HYBRID II - BBGS ALGORITHM 100
TABLE 6-26 RESULTS OF SCENARIO 4 FOR HYBRID II - BBBC ALGORITHM 101
TABLE 6-27 RESULTS OF SCENARIO 4 FOR HYBRID III - BBGS ALGORITHM 102
TABLE 6-28 RESULTS OF SCENARIO 4 FOR HYBRID III - BBBC ALGORITHM..................................... 102

CHAPTER 1

INTRODUCTION

1.1 Problem Definition

The problem of optimally assigning air force resources within an operation has been the focus of this thesis. Intelligent usage of these resources definitely provides huge advantages since the number of air force resources is limited relative to ground based resources and air force resources may play incredible and unique role in the battlefield.

Preparation of a flight is a costly job since it requires the efforts of many people, resources like fuel, and maintenance among many others. Therefore, careless usage of air force resources causes a great cost both in military and civil applications. In order to avoid such undesired costs, the operation to be conducted should be planned beforehand.

In civil transportation, the optimal assignment of fleets to destinations has been widely investigated, since it provides the means of reducing the cost of transportation, thus increasing the overall profit. This problem arises in military applications as well, even as a more important problem. In military applications, optimal assignment of resources is searched, not only for reducing the transportation costs, but also for managing at best in the matters of national security. The plan of an air operation in NATO is referred as ATO (Air Tasking Order) which includes the plan of operation and the match between resources and opposite side's components. A similar version of ATO also exists in Turkish military literature [13].

In this study, the optimal assignment of air force resources to ground based targets has been searched. The optimal assignment is said to be the one with the maximum profit. In this study, the searched assignment is not between aircrafts and targets, but instead between fleets and group of targets. A group of targets is referred as "chain" and it defines a collection of targets which are connected to each other in terms of operational meaning and precedence. The concept of chain is going to be explored in details in the upcoming sections.

The reasons for searching the assignment between fleets and chains instead of aircrafts and targets base on how the operations are conducted in real life. In real operations, to be able to fulfill the task on a target, it is generally required to fulfill tasks on other individual targets. For instance, to be able to hit a military headquarter, it is required to hit the missile weapon system and the radar controlling it beforehand for a safe flight. This is why the chain concept is introduced. Moreover, in operations aircrafts registered to different fleets are not assigned to the same target as long as there are sufficient aircrafts in the fleet to fulfill the task. Therefore, elevating the problem on fleet level does not harm the constraints of the problem. By elevating the problem to higher level provides the opportunity to search the solution via the same constraints but on a smaller solution space.

Optimal assignment between fleets and chains is said to be the one, maximizing the profit. The profit is defined with incomes and costs within the resource limits. It is desired to assign to the most valuable chains (each chain has a related value indicating its importance in the operation) while the total distance between assigned fleets and chains are attempted to be reduced as much as possible. Through this assignment process, only the feasible matches are valid; since the assets of a certain fleet may be insufficient to meet the requirements of some chains. These requirements of chains and assets of the fleets are going to be discussed in details soon.

For a better understanding of the problem, a small example scenario is given below. The circumstance given represents a very small problem which can be solved intuitively. Since an assignment satisfying the requirements of the chains and targets should be made, Fleet 2 cannot be assigned to Chain 1 and 42 because of Electronic Counter Measure (ECM) requirement and number of aircraft requirements. Fleet 1 has necessary conditions and aircrafts to be matched with Chain 1 and 42. Therefore the resulting assignment would be {(1, 1) (15, 2) (42, 1)}.

Table 1-1 A Simple Scenario for Representing the Problem

Scenario Chains & Targets within				
Chain Id	**Target Ids**	**#Aircrafts Required**	**ECM Required**	**RWR Required**
1	5 6 12 23 8	23	True	True
15	75 78	19	False	False
42	45 46 48 50 55	8	True	False
Fleets & Aircrafts within				
Fleet Id	**Aircraft Ids**	**#Aircrafts Possessed**	**ECM Equipped**	**RWR Equipped**
1	…	32	True	True
2	…	20	False	True

In the literature, there are similar problems researched, in which the assignment and planning of air resources in military operations has been investigated. UAVs dynamic assignment during the operation is one of them in which the total assignment is maximizing the score [12], [24]. The airlifting of military resources, which involves the assignment and scheduling part of the operation, has been explored in [11]. The similar problem has been investigated in [13]. Necessary data for information-based modeling of the operation; including the data of fleets, targets, aircrafts and chains, has been supplied [13].

1.2 Assignment Problems and Applications

As it was mentioned before, the problem is an assignment problem. Therefore, it is helpful to analyze the assignment problems and their important parameters. The assignment problem formulizes the objective and the constraints for finding the best assignment between tasks/jobs/locations etc and agents/objects/resources etc. Assignment problem may request to minimize the expenses of the assignment or to maximize the overall profit. Although the formulations of the assignment problems are investigated later, exploring some important properties of assignment problem may give insight to understand the objective of this study.

The most important parameter about assignment problems is the size of the problem. The size of an assignment problem is the number of elements that are going to be matched. As the size of an assignment problem increases, solving the problem gets harder since the exact solution methods fail to return a solution due to computational limitations. In such cases, the solution can only be found by heuristics. The solution returned by heuristics is generally not the optimum, but it is a good solution.

In practice, many sophisticated discrete optimization problems are assignment problems. Some examples of assignment problems are given below. The optimal assignment of units to maintenance problems is one of them [4]. The airport gate assignment where gates are assigned to scheduled flights requiring that the total distance the passengers walk is minimized, has been researched [5]. Moreover, locomotive assignment problem [6], frequency assignment [8], radio channel assignment problems [7], facility layout problems [9] are also some applications of assignment problems. As one of the most common and important application of the assignment problems is task assignment in distributed systems [10].

1.3 Aim of the Thesis

It is aimed to solve computationally difficult assignment problem which is described earlier by the specific case study. The aim is to solve the problem as fast as possible. For that purpose, the known algorithms are implemented and tested. Then, attempts have been made for designing and testing the means for improving the results of the known algorithms.

The components and information related to the case study, which are used for formulation and during the solution process, are modeled parameter-wise. Through the specific case study, the first goal is to identify the performance and applicability of widely known algorithm and heuristics. The basic algorithms are chosen as the Branch-and-Bound and Genetic algorithms.

Branch-and-Bound algorithm creates and searches a decision tree on which assignments or partial assignments are denoted by nodes and nodes are interconnected with each other by branches for referencing the hierarchical structure. Nodes leading to the parts of the solution space in which it has low or no probability to have a solution are pruned. In that way, an intelligent navigation and search on the solution space is performed. In its generic form, the algorithm starts from a root node on which no assignment is present. Then, algorithm branches to children nodes by dividing the solution space into subspaces and problem to subproblems. Although Branch-and-Bound algorithm is an exact solution method and provides tools necessary for the intelligent solution space search, it becomes computationally expensive as the size of the problem increases. The reason is that the size of the search tree increases exponentially with the problem size. As Branch-and-Bound algorithm is solely applied to the problem, it has been observed that this algorithm cannot suffice alone for this big problem due to memory requirements. Algorithm is halted before the global optimum is obtained, since there is no free memory left.

As Genetic algorithm is applied to the problem, with some problem specific heuristic methods imposing feasible solutions; it has been observed that the results are satisfactory. However, the returned solution by genetic algorithm depends on the randomly generated initial population. Moreover, since the solution found by Genetic algorithm is not the global optimum, this solution can be further improved by some other means.

Second aim of the thesis is defining and investigating the manners for further improving the results by proposing new methods which are efficient in spreading to the search space. For that purpose, the notion of distance has been introduced. The distance between two assignments (solutions) is the number of different elements between them. This distance concept enables to perform a neighborhood search around an assignment or a partial assignment. Distance idea has been applied to the Branch-and-Bound tree. Branches within a predefined distance (neighborhood) are considered as the scope of search. There are two proposed methods, which are using this distance measure for improving the solution. Both methods start with the Genetic algorithm. After certain iterations, the best chromosome of Genetic Algorithm is taken as the reference assignment. The first method keeps some elements of this reference assignment constant and uses this partial assignment as the root node of the Branch-and-Bound algorithm. Starting from this root node, remaining unassigned chains are processed by the Branch-and-Bound algorithm. The second method also uses the reference assignment coming from the Genetic algorithm. However, this method starts Branch-and-Bound algorithm from a root node where no initial assignment is present. The whole tree is processed, but only the branches within a certain distance (neighborhood) to the reference assignment are searched.

All the implemented algorithms are tested in various scenarios of the specific case study. These scenarios are selected and defined for testing the different properties of the algorithms. Scenarios can be classified as small size, medium size, and large size scenarios. Moreover, some scenarios impose loose constraints, while others involve tight constraints.

1.4 Outline of the Dissertation

Chapter 2 is describes the case study. The case study is defined and detailed with its properties. The extent of this study, based on the specific case study has been discussed.

Since the problem is an assignment problem, formulations of generic assignment problems are investigated for getting the inside on how to formulate such problems. Formulation of the case study is explained in Chapter 3. In this section, the related parameters on the fleet to chain assignment in military operations are introduced. The formulation of the problem is given and discussed in details.

Solution methods are illustrated in Chapter 4. The solution methods in the literature for assignment problems are explored. The reasons for selecting Branch-and-Bound and Genetic algorithm and the background information and introduction are given in this chapter as well.

All implemented algorithms are discussed in Chapter 5. The ways, Genetic algorithm and Branch-and-Bound algorithm is applied to the specific problem is expressed. Moreover, heuristics and altered forms of algorithms are proposed which are implemented and introduced for a possible improvement on the results.

Test scenarios and results are illustrated in Chapter 6. The created test scenarios are introduced. Later, the results of implemented algorithms for scenarios are given and commented. The performance of algorithms and their applicability to the specific problem are evaluated in this chapter.

Last chapter, Chapter 7, concludes the study. The work done is summarized. The success of this study is measured regarding what was aimed and what has been coped. The ways that this study can be further expanded, are discussed in the future work section.

CHAPTER 2

ON THE CASE STUDY

The corresponding assignment problem is aiming the optimal usage of air force resources. Of course, such statement is not sufficient for illustrating the problem. This chapter is dedicated for clearing all the obscurity about the assignment problem. All information on the specific case study, its parameters, nature and properties is given in the first section of the chapter.

As it was stated above the problem is to find the optimal assignment of air force resources to tasks. In the scope of this study the tasks have been restricted to ground based targets. At first glance, it may be predicted that the searched assignment should be between aircrafts and individual targets. However, it is not the case.

In real operation environment, it is a sophisticated task to discriminate the individual targets, since some targets may be connected to each other due to their dependence. This is generally true; a single target has elements around it for protection and they can also be considered as targets. Those targets play incredible role as well, since accomplishing task on the original target is practically impossible before taking actions to those targets. Therefore; in order to declare the operation of the original target as a success, the whole set of targets should be visualized as one target. In this study, the term “chain” is defined for that purpose. A chain is said to be a group of targets having relationship with each other. The figure below is for illustrating what has been meant by a chain.

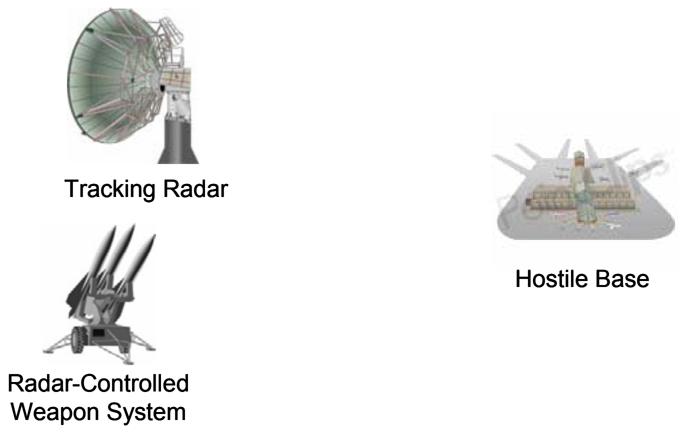


Figure 2.1 - Sample Chain

A sample chain has been given in Figure 2.1. The ultimate aim of the aircrafts is to hit the hostile base. However, the hostile base has been protected by a radar and a surface to air missile system controlled by the radar. The aircrafts cannot perform their jobs safely, if they discard the air defense units. Therefore; the whole hostile components must be regarded as one united target which is referred as a chain.

In most situations, the aircrafts for a particular task are selected from the same fleet as long as the fleet includes enough number of aircrafts loaded with equipments satisfying the requirements. The reason is to increase the coordination between the aircrafts. It is unreasonable to select aircrafts from different fleets, because briefing the staff of separately located fleets is not easy. Coordination internally in the fleet can be much easily performed.

Due to what has been mentioned above, the assignment problem in hand has been elevated from the aircraft-target level to the fleet-chain level. By doing so; not only the requirements of the real operations can be modeled truthfully, but also the size of the assignment problem has been reduced. By converting the problem to a fleet to chain assignment problem; the solution space is reduced, yet the constraints for the optimal assignment are somehow kept the same.

In order to formulate and solve the problem, some data about the components are necessary. The following subsections covers the details on the chains, fleets and how they are modeled parameter wise.

2.1 Chains

The way chains are considered and parametrically illustrated in the study has been discussed in this section. Moreover, the preprocessing for representing the chains is the other scope of this section. Before expressing the way chains are modeled, it is better to analyze targets, since a chain is formed by targets. Parameter wise, a target is explained as below.

Table 2-1 Representation of Targets

Field Name	**Description**
Target Id	Unique Id representing the target
Target Type	The type of the target
DMPI	Desired Mean Point of Impact
North Coordinate	North coordinate of the target
East Coordinate	East coordinate of the target
Impact Value	Impact value of the target
ECM	Electronic Countermeasures requirement of the target
RWR	Radar Warning Receiver requirement of the target
Required Aircraft Number	Number of Required Aircrafts of the target (Listed by aircraft type)

Each target has been given a unique Id. DMPI represents the number of critical points that the target has so that when aircrafts are assigned to this target, a certain impact should be satisfied for these points. Impact Value defines the desired amount of impact on a target. ECM and RWR parameters are for declaring that for an aircraft to be able to assign to a target having ECM requirement, then this aircraft should be

ECM equipped. ECM and RWR devices are self-protection systems of the aircrafts against hostile radars and weapon systems.

For each target, a certain number of aircrafts is required. A target's aircraft requirements depend on the aircraft type and its impact ratio, target's desired impact value, and number of desired impact points of the target. For a single DMPI, the number of aircrafts used for that impact point is kept increasing till the desired impact is satisfied. The total number of aircrafts is that number multiplied by the number of DMPIs. The algorithm for calculating targets' aircraft requirement is given below.

T	: List of Targets
#Aircraft Requirements	: The number of aircraft required of targets for all types of aircrafts
plane Impact	: Impact Ratio of an aircraft (depends on the aircraft type)

for $T_i \in T$

  for $i \in AircraftTypes\ \{1,2\}$

    $currentImpact = 0$

    $desiredImpact = T_i.ImpactValue$

    $planeImpact = (aircraftType == 1)\ ?\ 0.85 : 0.80$

    $usedAircrafts = 0$

    $DMPI = T_i.DMPI$

    while $(currentImpact < desiredImpact)$

      $usedAircrafts{+}{+}$

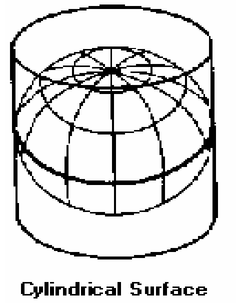
      $currentImpact = \left(1 - (1 - planeImpact)^{usedAircrafts}\right)$

    $T_i.\#AircraftsRequirements.Type.i = usedAircrafts \times DMPI$

$return\ T_i.\#AircraftsRequirements$

Algorithm 1 – Calculating the Number of Required Aircrafts of Targets

The location of the target is represented by its north and east coordinates. These coordinates are projected on the X-Y plane for providing ease in distance calculation. This conversion is given in Figure 2.2 [25].



$$x = \lambda$$

$$y = \phi$$

λ is the longitude from the central meridian of the projection and ϕ is the latitude

Figure 2.2 - Equirectangular Projection for Location Mapping

Four types of targets with different importances on the battlefield have been defined. Since each target type has its specific importance, they have specific values. Mostly in the operations, hostile bases and headquarters are severely important where the units protecting them like radars, defense units, etc. will have less importance comparatively.

Table 2-2 Target Types and Values

Target Type	**Inscription**	**Value**
Type 1	Hostile bases and headquarters	8
Type 2	Radars and Intelligence Units	6
Type 3	Air Defense Units and SAMs	4
Type 4	Others	2

As targets are defined, chains formed by multiple targets can be easily defined as well. In the extent of the study, a chain is restricted to contain at most five targets, yet this limitation is just arbitrary and is selected for convenience. The parameters of a chain are given in Table 2-3.

Table 2-3 Representation of Chains

Field Name	Description
Chain Id	Unique Id of the chain
Location	Location of the first target of the chain
Chain Value	The value(importance) of the chain
Target List	Ids of the targets involved in the chain
Num. of Required Aircraft	Total number of required aircrafts of the chain (Listed by aircraft type)
Has ECM	ECM Requirement of a chain
Has RWR	RWR Requirement of a chain

Each chain has been given a unique identification number. The location of the chain is used for distance calculation between fleets and chains. The location of the chain is said to be the location of the first target of the chain. The number of required aircraft of a chain is the sum of the number of required aircrafts of each target involved. A chain is said to have an ECM requirement if any target involved, has an ECM requirement. Similarly, a chain is said to have an RWR requirement if any target involved, has an RWR requirement.

Targets involved in a chain are given in the target list by their target Ids. In "Target List"; targets are ordered in time precedence. The first target in the list is the first one on the operational timeline. Sample target lists are given below.

Table 2-4 Sample Target List 1

Target 1	Target 2	Target 3	Target 4	Target 5
5	6	12	15	48

Table 2-5 Sample Target List 2

Target 1	Target 2	Target 3	Target 4	Target 5
21	26	1	-	-

As it can be observed from above, chains may have different numbers of targets involved in them. The importance of the target lists appears when the value of a chain is going to be calculated.

Chain value refers to the importance of chain in the operation. In this thesis, assigning to more valuable chains is kept more important than assigning to more chains. It is aimed to maximize the total value of assigned chains. The value of a chain depends on the targets within it, i.e. the target list. The last target in a chain is said to be the most important target in the chain, since the predecessor targets are for being able to safely perform the mission to this ultimate target. Value of a chain is calculated as the sum of the targets in it. As it was previously mentioned, targets have value according to their types. Weights have been given to the targets, related to their order in the chain. Weights of the sample target lists given in Table 2-4 and Table 2-5.

Table 2-6 Sample Target List & Weights 1

Target List	5	6	12	15	48
Weights	2	4	6	8	10

Table 2-7 Sample Target List & Weights 2

Target List	21	26	1	-	-
Weights	6	8	10	-	-

The last target of the chain is weighted by 10. Then, the others are weighted accordingly by descending weights of {8, 6, 4, 2}. Although the selected weights for chain calculation seem to be trivial, the weights are assigned so that minimum possible chain value and maximum possible chain value are still comparable. A chain’s total value is the superposition of targets within and calculated by the value of the targets involved and their position in the chain. A sample chain value calculation is as in the following table.

Table 2-8 Example of a Chain Value Calculation

	Target 1	Target 2	Target 3	Target 4	Target 5
Target Type	3	2	2	4	1
Target Value	**4**	**6**	**6**	**2**	**8**
Weights	**2**	**4**	**6**	**8**	**10**
Value	4x2=8	6x4=24	6x6=36	2x8=16	8x10=80
				Total:	164

2.2 Fleets

The description of fleets is not as complicated as it is for the chains. The parameters used for defining a fleet are given inTable 2-9;

Table 2-9 Representation of Fleets

Field Name	**Description**
Fleet Id	The unique Id specified for the fleet
Location	Location of the fleet
Aircrafts Type	Types of the aircrafts in the fleet
ECM	ECM capability of the aircrafts in the particular fleet
RWR	RWR capability of the aircrafts in the particular fleet
Aircraft Impact Ratio	Number representing the impact of the aircrafts in the fleet. for Aircraft Type 1; impact ratio = 0.85 for Aircraft Type 2; impact ratio = 0.80
Aircrafts List	Id of the aircrafts the fleet has

Each fleet has been given a unique identification number. The location of the fleet is necessary for distance calculation. ECM and RWR parameters are for indicating the ECM and RWR capabilities of the fleet. The type of the aircrafts in the fleet is defined. In the extent of this thesis, only two types of aircraft have been considered. Aircraft type plays an important role, since the impact ratio of the aircrafts is related to its type. Type 1 aircrafts has an impact ratio of 0.85, and second type aircrafts has 0.80. Aircraft impact ratio is used for calculating the number of required aircrafts of targets. The way, impact ratio is used, can be observed in Algorithm 1. Aircraft list stores the identification numbers of the aircrafts involved in the fleet. The number of aircrafts in the fleet is critical; because, for a particular chain, only the fleets having sufficient number of aircrafts are considered for assignment.

2.3 On the Extent of the Study

In the extent of the study, only 8 fleets are used. Since a fleet can be assigned to more than one chain, 8 fleets are generally sufficient to match the aircraft number requirement of the many of the chains. Moreover, it is assumed that aircrafts in a particular fleet are all of the same type.

The information on the fleets, chains, targets, and aircrafts are assumed to already exist. The way the priori information is gathered is not a concern in the scope of this study. The scenario of the operation has been retrieved from the database according to the defined data structure. The database also stores all the necessary data that may be useful in future use. Moreover, as the assignment of fleets to chains has been performed, the success of the aircrafts in the operation is not in the scope of the study as well. In this study, based on priori information on the components of the operation scenario, the assignment of air force resources (fleets) to group of targets (chains) has been performed.

ECM and RWR properties are included in the chains and fleets for inclusion of discrete Boolean constraints. By this way, it is aimed to state that, the number of this kind of constraints (ECM, RWR, night vision, etc) can be increased on wish.

As it was mentioned just earlier, in the database, all possible chain configurations and all possible targets are stored. This information is going to be used in the weight assignment of profit function.

Assignment has been done so that most profitable assignment can be found. In a case in which the number of chains is too high and all chains cannot be matched with a fleet, it is still aimed that among all, the most profitable and feasible assignment is done.

2.4 Classification of the Problem

The specific case study is better to be classified under a few types of topics. The assignment of aircrafts to targets within an operation is stated in the literature as assignment of air strike assets and investigated under study of “Air Tasking Order (ATO)”. ATO gives the plans of whole operation starting from the ordering, continuing with target assessment and asset assignment, and finally planning of the operation on a timeline possibly in multiple sorties. Researches on this subject is listed in [11], [13], and [33]. The case study of this thesis includes targets assessment and asset assignment in which only the planning of a single sortie (most profitable one) has been searched.

In the literature, there are some similar problems, in which the assignment and planning of air resources in military operations has been investigated. UAVs dynamic assignment during the operation is one of them in which the total assignment is maximizing the score [12], [24]. More studies on this subject are given in [32].

Weapon-target assignment (WTA) problems investigates the most profitable or least costly assignment of existing weapons to existing targets within certain constraints and a certain scenario. A sample WTA problem and its formulation are explored in [29]. Considering the specific case study of this thesis, the chains can be regarded as weapons. On the other hand, the formulation of a WTA problem is vastly different from the formulation of this study. Therefore, we can conclude that, although investigating WTA problems gives an insight to the problem in hand, it does not provide more for our problem.

CHAPTER 3

PROBLEM FORMULATION

The formulations of assignment problems are given for the purpose of better understanding of the problem type. Then, the formulation of the specific problem has been given.

3.1 Assignment Problems and Formulations

As one of the most-known and significant concerns of the combinatorial optimization, assignment problems have been studied and applied in many different applications. Since the specific case study is an assignment problem, the formulations of best known assignment problems are investigated. The formulations are done for a position object allocation problem, just for illustration.

3.1.1 Quadratic Assignment Problem

As a specific type of assignment problems, QAP has the property of modeling many of the applications such as task allocation, traveling salesman problem and scheduling problem. In most formulations, terms “distance” and “flow” have been used frequently. “Flow” defines the number of total commodities been used in a specific assignment, where “distance” stands for the penalty of the assignment in terms of distance. There are a few QAP formulations in the literature. However, the one providing the foundation of other formulations is Koopmans and Beckmann formulation. In this formulation, Boolean variables indicate whether an object and a location have been matched [1], [2].

$$\pi(i) = j \Rightarrow x_{ij} = 1 \Leftarrow \pi(j) = i \tag{3.1}$$

A Boolean variable, x_{ij} is 1 if object i is assigned to position j. Moreover, Boolean variables are relaxed with binary constraints which restrict the assignment to be one on one, where each object should be assigned to one location and vice versa. The objective function is as in equation (3.2).

$$\min \sum_{i}^{n} \sum_{j}^{n} \sum_{k}^{n} \sum_{p}^{n} f_{ij} \cdot d_{kp} \cdot x_{ik} \cdot x_{jp} \tag{3.2}$$

$$s.t.$$

$$\sum_{i,j=1}^{n} x_{ij} = 1 \quad 1 \le i, j \le n \tag{3.3}$$

$$x_{ij} \in \{0,1\} \quad 1 \le i, j \le n \tag{3.4}$$

The constraints of the problem are stated in equations (3.3) and (3.4). These constraints simply indicate that no objects can be assigned to more than one position and vice versa.

3.1.2 Linear Assignment Problem

Linear assignment problems are one of the mostly observed assignment problem and can be solved easily via Hungarian method in polynomial time for small size problem.

$$\min \sum_{i}^{n} \sum_{j}^{n} c_{ij} \cdot x_{ij}$$
$$s.t. \tag{3.5}$$

$$\sum_{i=1}^{n} x_{ij} = 1 \quad 1 \le j \le n \tag{3.6}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad 1 \le i \le n \tag{3.7}$$

$$x_{ij} \in \{0,1\} \quad 1 \le i, j \le n \tag{3.8}$$

The first thing that draws attention is in (3.5), this problem formulation requires no linearization, since the assignment decision variable x_{ij} stores just the necessary information about the assignment and it is linear. The objective of the problem has been given in equation (3.5). Constraints (3.6) and (3.7) state one to one assignment, which one object can only be assigned to one location and one location can be matched with only one object. (3.8) is for declaring that decision variable can only be zero or one.

3.1.3 Generalized Assignment Problem

Formulation of the generalized assignment problem of assigning n jobs to m agents while satisfying resource constraints for each agent has been formulated by Sagbansua as follows; [3]

$$\min \sum_{i}^{n} \sum_{j}^{n} c_{ij} \cdot x_{ij}$$
$$s.t. \tag{3.9}$$

(3.9) illustrates the objective function where (3.10), (3.11) and (3.12) are the constraints of the problem.

$$\sum_{i=1}^{n} x_{ij} = 1 \quad 1 \le j \le n \tag{3.10}$$

$$x_{ij} \in \{0,1\} \quad 1 \le j \le n, 1 \le i \le m \tag{3.11}$$

In equation (3.12), the resource constraints for the assignment are given. r_{ij} represents the resource required of assigning j^{th} job to i^{th} agent and b_i represents the available resources of the i^{th} agent. Assignments that do not satisfy the constraint below are said to be infeasible assignments.

$$\sum_{j=1}^{n} r_{ij} \cdot x_{ij} \le b_i \quad 1 \le i \le m \tag{3.12}$$

3.2 Problem Formulation

The problem can be specified as a resource-constraints assignment problem. The mathematical formulation of the problem is given below where m is the number of fleets and n is the number of chains.

$$\max \sum_{i=1}^{n} \sum_{j=1}^{m} p_{ij} \cdot x_{ij} \tag{3.13}$$

s.t.

$$\sum_{j=1}^{m} x_{ij} \le 1 \quad \forall i = 1, \ldots, n \tag{3.14}$$

$$\sum_{i=1}^{n} r_i \cdot x_{ij} \le b_j \quad \forall j = 1, \ldots, m \tag{3.15}$$

$$\sum_{j=1}^{m} fleetECM_j \cdot x_{ij} \ge chainECM_i \quad \forall i = 1, \ldots, n \tag{3.16}$$

$$\sum_{j=1}^{m} fleetRWR_j \cdot x_{ij} \ge chainRWR_i \quad \forall i = 1, \ldots, n \tag{3.17}$$

$$x_{ij} \in \{0,1\} \quad \begin{matrix} \forall i = 1, \ldots, n \\ \forall j = 1, \ldots, m \end{matrix} \tag{3.18}$$

$$\begin{aligned} &fleetECM_j \in \{0,1\} && \forall j = 1,\ldots,m \\ &fleetRWR_j \in \{0,1\} && \forall j = 1,\ldots,m \\ &chainECM_i \in \{0,1\} && \forall i = 1,\ldots,n \\ &chainECM_i \in \{0,1\} && \forall i = 1,\ldots,n \end{aligned} \tag{3.19}$$

In equation (3.13), p_{ij} is the profit of assigning fleet i to chain j. The profit of a single assignment is always assumed and set to be greater than zero. The equation of a single assignment profit is given in (3.20). As it was mentioned the previous chapter, each chain has been given a value regarding the targets within it. The income of an assignment is the value of the chain assigned and it is represented by v_i. Therefore, for maximizing the profit, chains with higher values should be assigned carefully. The profit may decrease by the distance between chain location and fleet location (d_{ij}); because it is not desired that the aircrafts fly large distances. The fleet to chain assignment with less distance should be preferred. As it can be seen from (3.20), the profit of a single assignment is a weighted sum of chain value (v_i) and distance (d_{ij}). Weight of the distance has been chosen as 0.001, because the distance is in meters. By multiplying it with 0.001, distance has been converted to kilometers. As it was stated earlier, all possible chain configurations and targets are stored. This information is used for selecting the weight of the chain value, so that the profit of a single assignment can always be larger than zero.

$$p_{ij} = 70 \cdot v_i - \left(1/1000 \right) \cdot d_{ij} \tag{3.20}$$

Equations from (3.14) to (3.19) are the constraints of the problem. Equation (3.14) states that a chain can be assigned only to one fleet, yet it may be assigned to neither as well. On the other hand, this is not true for fleets; a fleet can be assigned to many chains as long as it possesses the necessary resources. In this equation, x_{ij} is the decision variable where it is equal to one when fleet i is matched with chain j.

Equation (3.15) guarantees that the total number of required aircrafts of all chains matched with a specific fleet cannot be more than the number of aircrafts available in the fleet. In this notation, r_i is the number of required aircraft by chain j. "b_i" is the total number of aircrafts available in the fleet i.

What has been stated in equation (3.16) is as follows. For a fleet to be assigned to a chain, if the chain has an ECM requirement than the fleet must have aircrafts equipped with ECM devices. On the other hand, if the chain does not require an ECM, then it does not matter whether the assigned fleet has ECM capability or not. Equation (3.17) is the RWR version of this constraint.

The formulation of the problem reveals the fact that the problem is a linear problem. The objective function and constraints are linear, yet most of the variables are restricted to be zero or one. Considering these facts; it can be expressed that the problem is an integer linear programming problem.

In Chapter 1, it has been stated that the most important parameter of an assignment problem is the size of the problem. It was also stated that as the size of the problem increases, the exact solution of the problem cannot be found due to limited computation capacity. At this point, constraints become extremely critical since they indicate the feasible parts of the solution space.

CHAPTER 4

SOLUTION METHODS

The most known solution techniques for solving large scale assignment problems are expressed in this section. Among all, mostly preferred methods and algorithms are mentioned. In this study, Branch-and-Bound algorithm and Genetic algorithm are selected for implementation. The reason for such a selection has been discussed.

4.1 Solution Methods in the Literature

Assignment problems can be solved by mathematical means and methods. Linear assignment problems can be simply solved by Hungarian method. Brute force solution method and linear programming means are also legitimate for these problems. Methods like Branch-and-Bound, dynamic programming, and cutting plane technique are exact solution methods. However, for problem size larger than 15, these exact solutions and mathematical means fail to obtain the solution due to memory and CPU requirements. For those cases, heuristics are used like Genetic algorithms, simulated annealing and tabu search. Those algorithms are discussed below.

4.1.1 GRASP (Greedy Random Adaptive Search Procedure)

This iterative technique starts with an initial feasible solution. The search has two phases where in the construction phase feasible solutions are created. In the second phase, improvement phase, a local neighborhood search is performed. The best solution after a certain number of iterations is said to be optimum [14], [17].

4.1.2 Ant System

This method introduced by Dorigo is based on the moves of the ant colonies. Ant agents move between discrete states or solutions and as they move, a shared pheromone matrix is updated. This matrix illustrates the desirability for all assignments. Agents (ants) move to those solutions which are more desirable [14], [15], [16], [17].

4.1.3 Tabu Search

Tabu search is an iterative search procedure. The word "tabu" is used, since some alteration on the current solution are said to be tabu moves because they lead to worst solutions. At each iteration, candidate neighborhood moves are evaluated which lead the current solution to a new solution. Restrictions are imposed to classify certain tabu moves, thus those moves are either discouraged or forbidden. The applications of this technique to assignment problems are common [14], [15], [17], [18].

4.1.4 Simulated Annealing

Inspired by the annealing of the metals; this algorithm creates a more solution in each iteration. Gradually cooling of metals makes them tougher. For a minimization problem, algorithm seeks to decrease the objective function. However, for avoiding local minimum, algorithm also attempts to increase the objective function occasionally [14].

4.1.5 Branch-and-Bound Algorithm

The algorithm creates and searches a decision tree on which assignment or partial assignment are created as nodes and nodes are interconnected with each other by branches for referencing a hierarchical structure. On the tree, nodes leading to the parts of the solution space in which it has low or no probability to have a solution are pruned. In that way, an intelligent navigation and search on the solution space can be performed. Studies on Branch-and-Bound algorithm are given later.

4.1.6 Genetic Algorithm

As an algorithm inspired by the reproduction process in the nature, in genetic algorithm, a certain portion of the solution space is mapped to chromosomes. As certain number of iteration has been done, in each selection, crossover and mutation operators are performed, the fittest chromosome in the population is said to be the solution. Studies on Genetic algorithm are given later.

4.2 Selected Methods

Besides solving the specific assignment problem of optimal usage of air force resources in operations, the aim of the study is to investigate the efficient search and solution algorithms. As the problem is being solved, it is also desired to find, define, and analyze the intelligent manners of solution space exploration. For that purpose, a couple of known algorithms should be selected and heuristics for improving the results should be applied. Therefore, the algorithms which are serving this purpose most are selected for implementation.

Branch-and-Bound algorithm and Genetic algorithm are considered convenient for the purpose. Since it is an exact solution method and has been based on a systematic and hierarchical structure, Branch-and-Bound algorithm has been selected as the first algorithm. Branch-and-Bound algorithm searches the solution by generating a solution tree. On the tree, directly or indirectly all the elements of the solution space are somehow processed. On the other hand, for large problems whose size is larger than 15, the solution space is too big and the algorithm faces computational problems.

Genetic algorithm, on the other side, is very computationally efficient. A certain number of solutions are mapped to chromosomes to create the initial population.

Starting from this initial population, by special operators, better solutions are searched. Since the number of processed solution in each iteration is constant. The computational difficulties are not a problem in genetic algorithm. On the other hand, genetic algorithm most probably does not return the global optimum. Moreover, the success of the obtained solution depends on the randomly generated initial population and the efficiency of the operators.

Branch-and-Bound algorithm and Genetic algorithm are used together, because considering their advantages and disadvantages of each one; they seem to be complement of each other. Weakness of the Branch-and-Bound algorithm due to computational difficulties can be overcomed by previously running Genetic algorithm and using the information provided by it in the search of Branch-and-Bound algorithm.

4.3 Branch-and-Bound Algorithm

In this section, the general form of the Branch-and-Bound algorithm has been introduced. The general structure of the algorithm, search strategies, and implementation concerns are discussed. The use of the algorithm in the literature has been expressed.

4.3.1 On Branch-and-Bound Algorithm

A thorough survey and instructive work on Branch-and-Bound algorithm has been done by Brixius [14] and Chinneck [35]. The following information has been based on those studies.

Branch-and-Bound algorithm is the main solution method for solving discrete integer programming problems. It enumerates to solutions on a tree structure. It works systematically and intelligently avoiding the tree growing too much to handle.

Branch-and-Bound algorithm is to date the most effective exact solution procedure for assignment problems. Branch-and-Bound algorithm solves the assignment problems by performing partial assignments and evaluating (computing the lower bound of the partial assignments) them. By doing so, most of the elements of the solution space have been eliminated from evaluation. Therefore, the solution is obtained in a shorter process time.

The basic idea behind Branch-and-Bound algorithm is the tricks on the solution space and its subspaces. What Branch-and-Bound algorithm does is to divide the original solution space into subspaces where the solution has been systematically searched, in each of them by relaxing the constraints (creating subproblems) of the original problem. For this purpose, in Branch-and-Bound algorithm, a tree structure is created and nodes are generated to represent the subspaces and subproblems. Each node simply holds the necessary information enough to illustrate the subspace it is representing.

In assignment problems, Branch-and-Bound algorithm starts with the root node which may refer to a partial assignment or no assignment at all. Branching operation is used for selecting the children nodes (subspaces) and creating them. In a branching operation, either single assignment is fixed or a collection of single assignments are fixed. Efficient branching has a great role in this algorithm. In the extent of this thesis, single assignment branching has been chosen regarding the ease in visualization and implementation concerns.

Speaking of a generic Branch-and-Bound tree, in each node, an equivalent of sub problem is solved by relaxing the constraints of this sub problem where this solution corresponds a lower bound of the original problem. The overall best solution is said to be incumbent and this solution must be a feasible solution. On the other hand, bound on a node does not necessarily refer to a feasible solution, because it is the solution to the relaxed version of the original problem. The bound is an estimator on the real problem. Mostly, bounding function is the solution of the problem which is generated by ignoring some constraints of the original problem. As bounding function gets better, which is to say that a better estimate on the original problem is found, the size of the tree gets smaller. Feasibility checks and bounding functions play incredible role in Branch-and-Bound algorithm, because these two concepts mainly define the efficiency of the algorithm.

In cases in which the lower bound calculated at the node is greater than the incumbent (for a minimization problem), then the node and its successor nodes are not going to lead to a solution. Therefore, the node is fathomed (pruned) which minimizes the size of the tree. For that reason; a priori incumbent value will definitely increase the efficiency, because more nodes can be pruned. Children nodes are then created for those non-pruned nodes. As the tree is constructed and examined, the optimal solution is said to be found because all nodes are explored directly or indirectly.

One of the most important concepts of Branch-and-Bound algorithm is the node selection strategy. Although it has no effect on the final solution itself, this strategy plays a great role on the efficiency of the algorithm, generally related with the specific problem and its requirements. Three main selection strategies are mainly stated.

- Breadth-First Strategy: The node that has been in the "to be processed" list the longest, is chosen. This search is disadvantageous since the tree enlarges exponentially and list of nodes to be processed becomes extremely large to be manageable. This technique searches the tree horizontally.
- Depth-First Strategy: Unlike from breadth-first strategy, depth-first search selects the node that has been recently added to the list. Relative to breadth-first, the size of the list is smaller and it is easier to operate with this list. Computationally, the algorithm is simpler and understandable.
- Best-First Strategy: this strategy selects the node returning the minimum or maximum value of a certain criterion function. The return value of the criterion function is an estimate on the value of the newly generated sub problem. By this way, it is aimed to lessen the time to attain the optimal. On the other hand, since the list of nodes should be ordered, the process takes more CPU time.

As it was mentioned in the earlier sections, the drawback of Branch-and-Bound algorithm is that at some point the solution tree grows too much and becomes computationally unmanageable. The size of the tree increases as well within an exponential relation with the size of the problem. Therefore, for large scale problems, optimal solution cannot be obtained due to computational limitations.

4.3.2 Branch-and-Bound Algorithm for Assignment Problems

Brixius has studied on the generic problem of solving QAP with Branch-and-Bound algorithm and the alternative branching and bounding methods has been stated [14]. As it was mentioned before, his study is also an instructive reference on this literature. More applications of Branch-and-Bound algorithm to generic assignment problems can be found in [19], [20] and [21].

In [22], Branch-and-Bound algorithm has been applied for finding the optimal scheduling of thermal generating units. Somol, Pudil, and Kittler has used Branch and Bound algorithm for selection of the minimal sufficient set of feature for a specific purpose [23]. A similar application, to the study of this dissertation, of Branch and Bound algorithm can be seen in [12] and [24].

Previously mentioned, assignment of utility systems problem has also been solved by Branch-and-Bound algorithm [4]. Moreover, the locomotive assignment problem has been solved as well in [6].

4.4 Genetic Algorithm

As an algorithm inspired by the evolution of organism, in Genetic Algorithm, each chromosome is a member of the population and represents a solution from the solution space. Starting from an initial population, by performing some operators adapted from again the evolution process, it is attempted to add better individuals to the population. These operators are reproduction, crossover, and mutation. The algorithm is based on the fact that after a certain number of evolution cycles, the best (fittest) chromosome in the population will be the desired solution, at least a preferable one.

4.4.1 On Genetic Algorithm

Genetic algorithm is advantageous in many ways. It does not require deep knowledge on the problem and the algorithm is easy to implement. The flow of the algorithm is given in Figure 4.1.

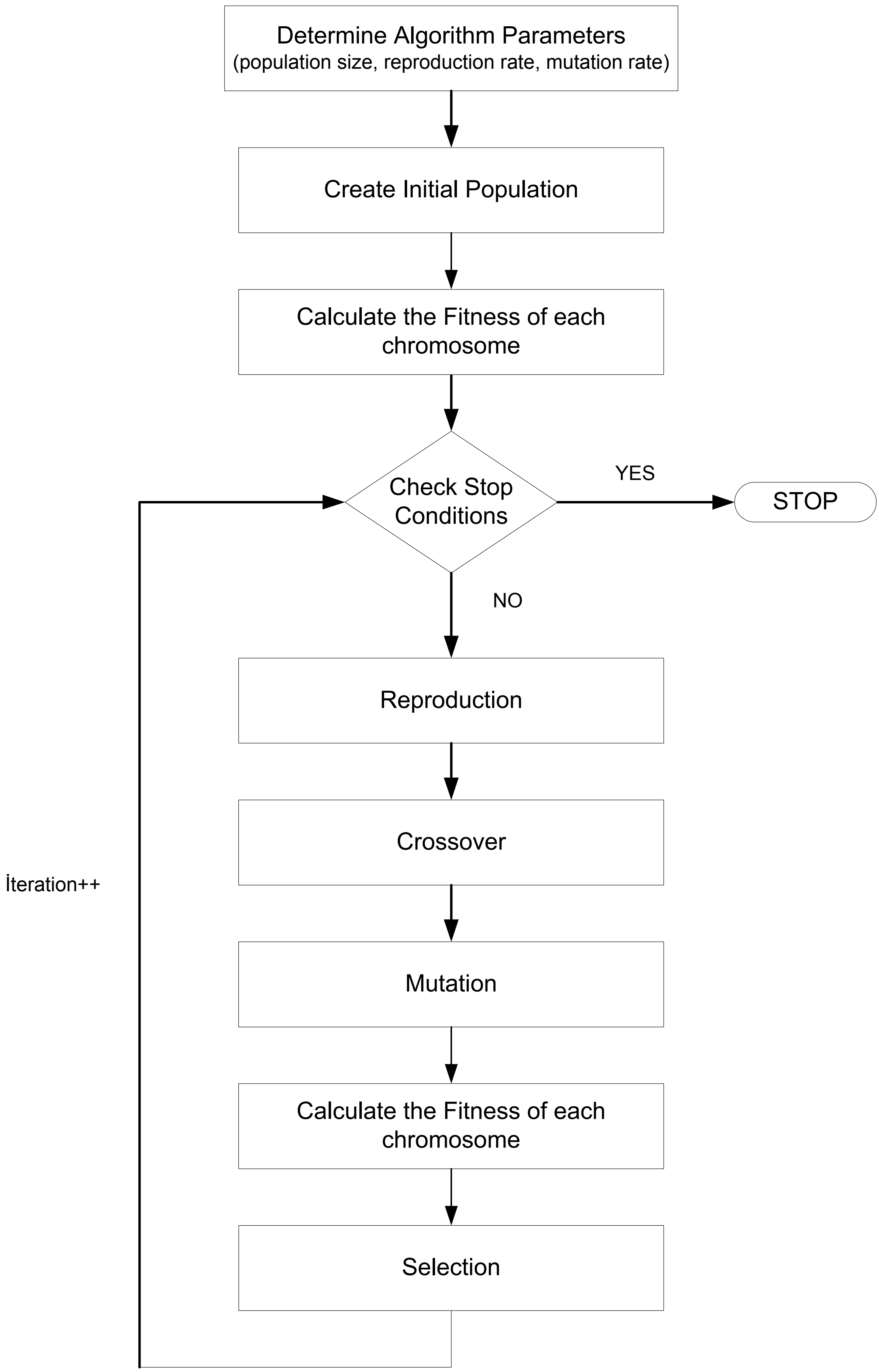


Figure 4.1 - Flow Diagram of a Generic Genetic Algorithm

The algorithm starts with the design. The size of the population and stop conditions of the algorithm are determined. Then, the reproduction rate is determined. Reproduction rate is used for preparing the mating pool. Population size multiplied by reproduction rate gives the number of elements in the mating pool which are going to evolve to children. The mutation rate defines how often mutation operator is going to be used in an iteration [35].

As the parameters of the algorithm have been determined, the initial population is created. In most cases, the initial population is selected randomly. On the other hand, the quality of the solution of Genetic algorithm severely depends on the quality of the initial population. After the initialization of the population, fitness of each chromosome in the population is calculated. Fitness of a chromosome indicates the strength of the solution which is its potential of being a good solution among the population. A fit chromosome is said to refer a better solution.

The operators mentioned in the flow are detailed as so;

- **Reproduction:** The chromosomes which are going to be added to the mating pool for reproduction are selected by this operator. The chromosomes selected to the mating pool are used for creation of new chromosomes by crossover operation. There are two most preferred reproduction methods in the literature. The first one is random selection; the chromosomes are selected without regarding their fitness or any other properties. The second method is Roulette Wheel Selection method; fittest chromosomes are most probable to be selected.
- **Crossover:** Crossover operation is used to create new chromosomes by mating the chromosomes in the mating pool. There are many crossover methods, each of which may serve different purposes. Two main and commonly known crossover methods are single point and two point crossovers. These methods are illustrated in the figure below.

AAAAAA AAA → AAAAAA BBB

BBBBBB BBB → BBBBBB AAA

Figure 4.2 - Single Point Crossover

AAAAAAAAAA → AAAABBBAA

BBBBBBBBBB → BBBBAAABB

Figure 4.3 - Two Point Crossover

- **Mutation:** Although crossover operation creates new chromosomes for the population, those newly created individuals may be not fit enough to survive in the population. This situation causes the population to stuck on a state where the population stops improving and a local optimum point emerges as a global point. In order to avoid such cases, mutation operator is used. At a random ratio, this operator alters some chromosomes' information to convert them to alternative chromosomes. By doing so, variety in the population is provided.

- **Selection:** Application of the selection operation or the way it is applied is totally optional. At the end of iteration, the size of the population is increased due the creation of new chromosomes by crossover. Generally, an elitist selection is used in which the elitist members of the population (fittest ones) are selected to carry on the new iteration. The remaining members are deleted.

4.4.2 Genetic Algorithm for Assignment Problems

Since Genetic algorithm is applicable to many problems, its use in the literature is dense. However, in this section, the types of assignment problems which are solved via Genetic algorithm have been discussed.

In [31], a generic application of Genetic algorithm to generalized assignment problem has been investigated. Efficient synthesis of finite state machines and optimal state assignment problem is solved by Genetic algorithm for the purpose of fewer area and delays [27]. Task assignment in distributed systems [28] and frequency assignment problem [30] are also sample applications in which Genetic algorithm is chosen as the method of solution. In [26], Genetic algorithm has been used for increasing the capacity of air traffic by traffic assignment of aircrafts.

Genetic algorithm is used for the assignment of military units as well. The most common application is Weapon-Target Assignment (WTA) problem. The problem in the scope of this dissertation can also be regarded as a weapon-target assignment with a very specific formulation. A WTA problem is studied in [29], in which battle formations are arranged for efficient air defense. In [32], task assignment of UAVs is investigated. Assigning multiple UAVs to multiple tasks are performed.

CHAPTER 5

IMPLEMENTATION AND HEURISTICS

All implemented algorithms and heuristics are discussed in this chapter. Although it is attempted to implement the algorithms as close as possible to their generic forms, many alterations on the algorithms have been performed for making them suitable for the specific case study problem. The main points of the case study and the formulation of the problem have been given in Chapter 2 and Chapter 3, respectively. In implementation, equations (3.14), (3.15), (3.16), and (3.17) are used for feasibility check of an assignment. The profit of the assignment is calculated by equation (3.13).

In Chapter 4, it has been stated that Branch-and-Bound algorithm and Genetic algorithm are selected as the source algorithms in this study. These algorithms in their specified forms for the specific problem are implemented. Then, the algorithms are combined with various heuristics and with each other intending an improvement in their efficiency. All implemented algorithms can be classified under five main titles. All algorithms are discussed in the following subsections; application and implementation details for the specific problem. Implementation has been done in C++ programming language. The implemented algorithms are classified as so;

- Branch-and-Bound Algorithm (BB)
  - Branch-and-Bound - Global Search (BB-GS)
  - Branch-and-Bound – Bounding Case (BB-BC)
- Genetic Algorithm (GA)
- Branch-and-Bound – Bounding Case with priori information by Genetic algorithm (BB-BC + GA) → HYBRID I
- Branch-and-Bound algorithm initiated from Genetic Algorithm partial assignment → HYBRID II
  - GA + BB-GS with Partial Assignment
  - GA + BB-BC with Partial Assignment
- Branch-and-Bound with neighborhood search initiated from Genetic Algorithm full assignment → HYBRID III
  - GA + BB-GS with Neighborhood Search
  - GA + BB-BC with Neighborhood Search

5.1 Branch-and-Bound Algorithm

Branch-and-Bound algorithm has been introduced in Chapter 4. It can be observed that the algorithm bases on the elimination of unnecessary parts of the solution space by bounding criteria and feasibility check. In order to observe the efficiency of the bounding function, two separate form of the algorithm are applied. In the first one, no bounding function is used; pruning is based on only the feasibility check. This version, we would like to call as "Branch-and-Bound – Global Search". In the second version, a bounding function has been included. Although this one is the generic form of the algorithm, in this study, for a better discrimination, it is called as "Branch-and-Bound – Bounding Case". As it may be remembered from the previous chapter, there are tree search strategies in Branch-and-Bound algorithm. These strategies are Breadth-First, Depth-First and Best-First search strategies. Each of "Branch-and-Bound – Global Search" and "Branch-and-Bound – Bounding Case" is implemented for these tree search strategies.

Before getting into the details of the two form of the algorithm, it is better to identify a node on a tree, because all the information and processing is carried on nodes. A specific node structure has been designed for this particular problem. This node structure is as follows.

Table 5-1 Node Structure of the Tree

NODE		
Variable	**Data Type**	**Explanation**
Node Id	Integer	Unique Id for referring the specific node
Profit So Far	Float	Overall profit of the assignment fixed so far
Depth	Integer	The depth of the node on the search tree
Assignment	vector <Single Assignment>	Assignment so far (Single assignment: short Chain Id, short Fleet Id)
Fleet Aircraft Numbers	Array[short]	Remaining aircraft numbers of fleets from the assignments
Bound	Float	Bound calculated for the node (each node refers to a sub problem) valid for BB-BC
Distance	Integer	Valid for neighborhood search on the tree (will be soon explained)

5.1.1 Branch-and-Bound – Global Search (BB-GS)

In BB-GS (Branch-and-Bound – Global Search), no bound calculation has performed. Pruning is based on feasibility check. There is no assignment on the root node. A sample search tree for this specific problem is given in Figure 5.1 for illustration purposes. How the Branch-and-Bound algorithm processes is investigated and explained through this sample tree. In this sample, it is assumed that there exist three fleets having different numbers of aircrafts.

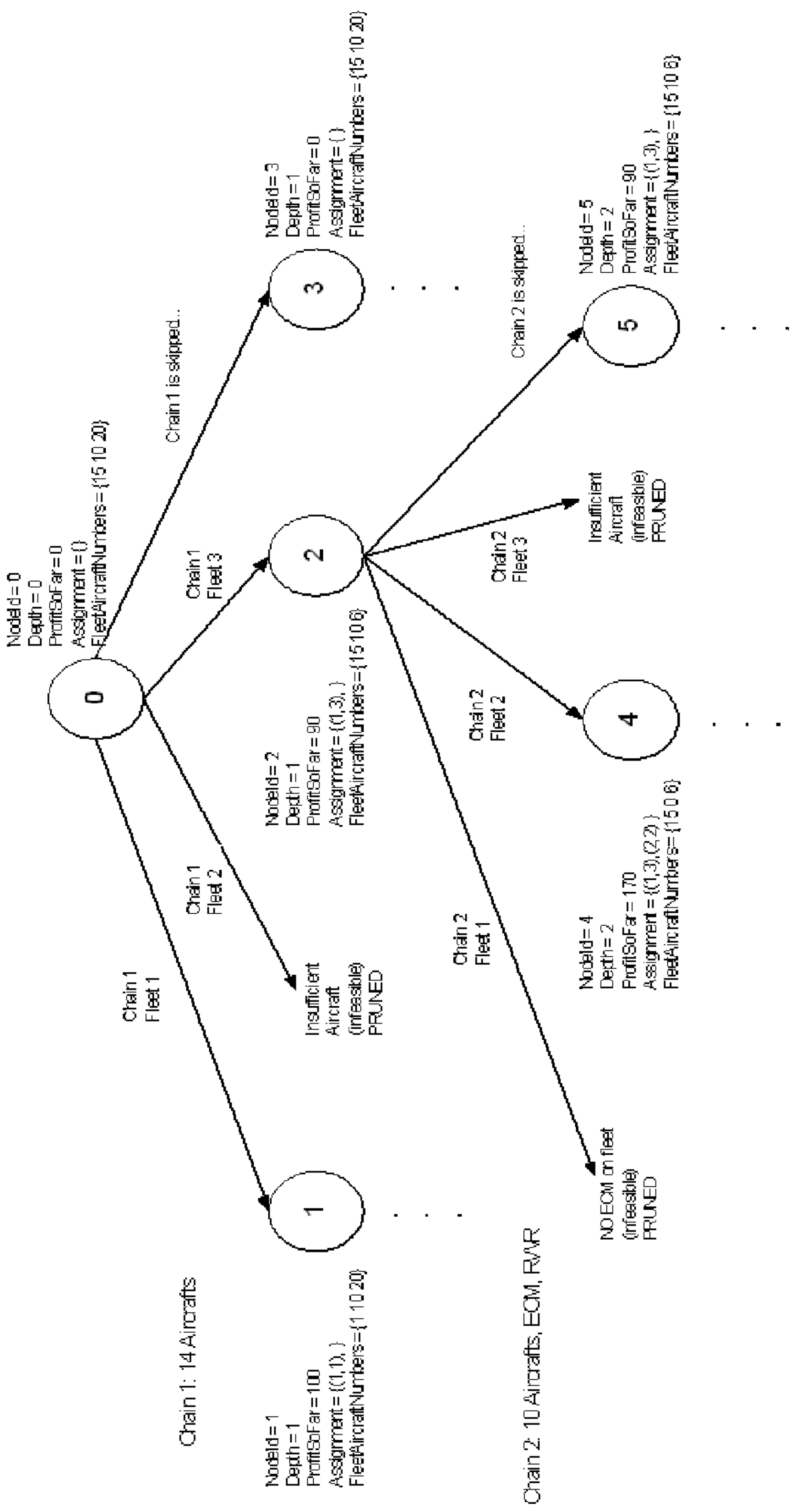


Figure 5.1 - Branch-and-Bound – Global Search: Sample Search Tree

As it can be followed from the figure, algorithm starts with the root node where no profit has been done yet. Since single assignment strategy is chosen to be applied, branching is performed for a specific chain. In this thesis, chains are sorted in terms of their value and assignment has started from the most valuable chain and so on. Children nodes of the current node, whose number is as much as the number of assignable fleets, are checked for feasibility. Feasible children are created and their profits are calculated. Resources, aircraft numbers in fleets, are updated since the number of aircrafts of the assigned fleet should be reduced. In the search, it may be also desirable to skip assigning the chain to any fleet since in some cases; assigning two chains may be more profitable than assigning one important chain. Resources may be used for assignments of other chains. Another topic worth mentioning is that some nodes and their successors are pruned. As it was previously stated, no bound related pruning has been done in BB-GS. Nodes are pruned because of three circumstances, which represent the feasibility check.

- No sufficient aircraft exists.
- ECM requirement does not satisfy.
- RWR requirement does not satisfy.

All of the tree search strategies are implemented in BB-GS. Effort has been given to efficient implementation, because the problem is large and computationally expensive. The memory and CPU time concerns are minimized.

5.1.1.1 Breadth-First Strategy (BreFS)

Breadth-First search strategy scans the tree horizontally. The flow diagram of this technique has been given in the figure below.

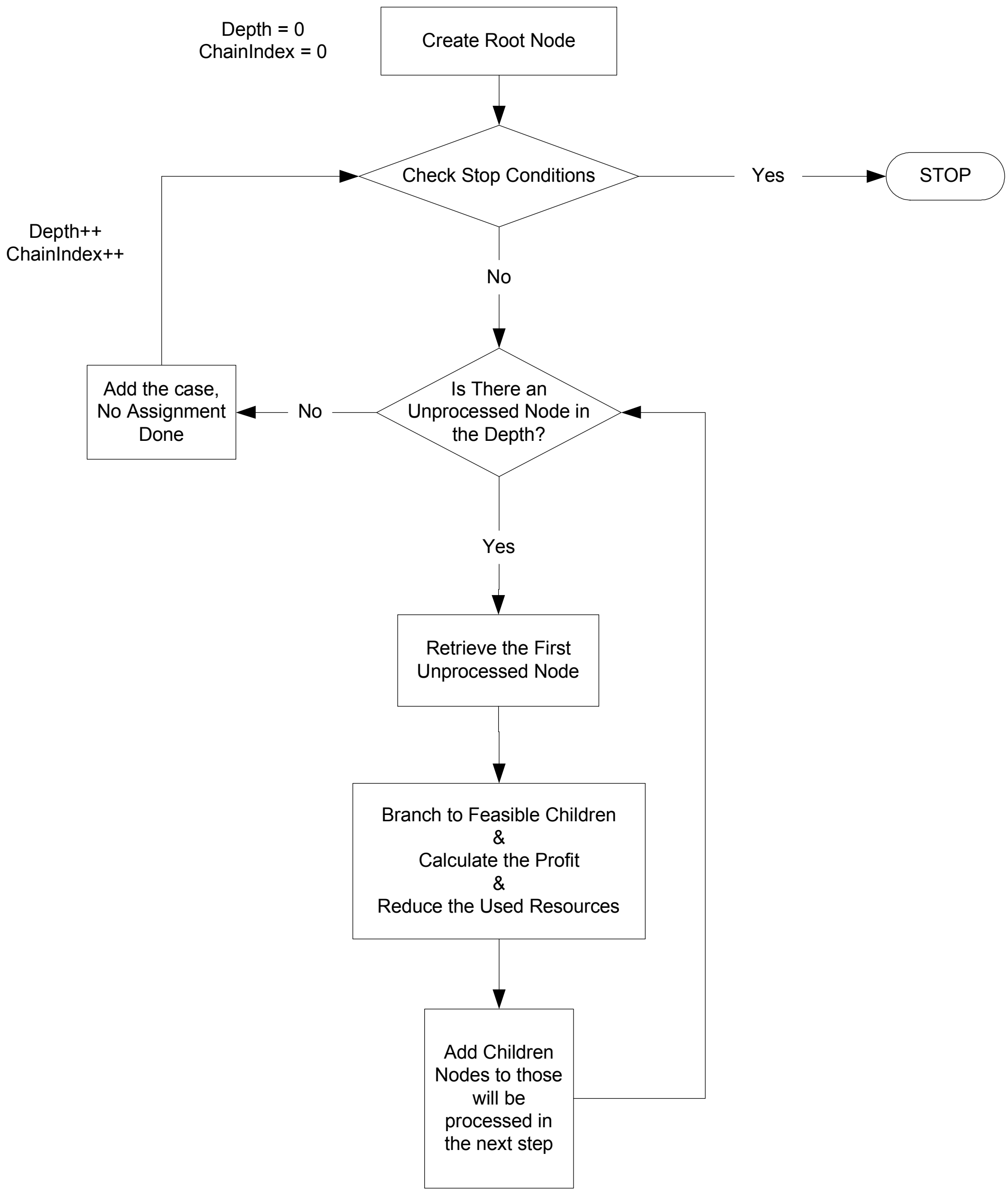


Figure 5.2 - Flow Diagram of BB-GS: Breadth-First Search Strategy

In this search strategy, in each depth of the tree a chain is processed and assignments for that particular chain are made. Algorithm does not operate on the next depth until all the nodes in the current depth are processed. All children nodes originating from a certain depth are added into a queue of unprocessed nodes. The first element of this unprocessed node list is going to be investigated first in the next depth. For cases where few nodes are pruned, this search strategy faces memory difficulties too soon. When large problems are attempted to be solved by Branch-and-Bound algorithm in a machine having not enough memory, this algorithm will halt in the middle of the search due to insufficient memory and returns no solution.

5.1.1.2 Depth-First Strategy (DFS)

Depth First search strategy scans the tree vertically. The flow diagram of this technique has been given in the figure below.

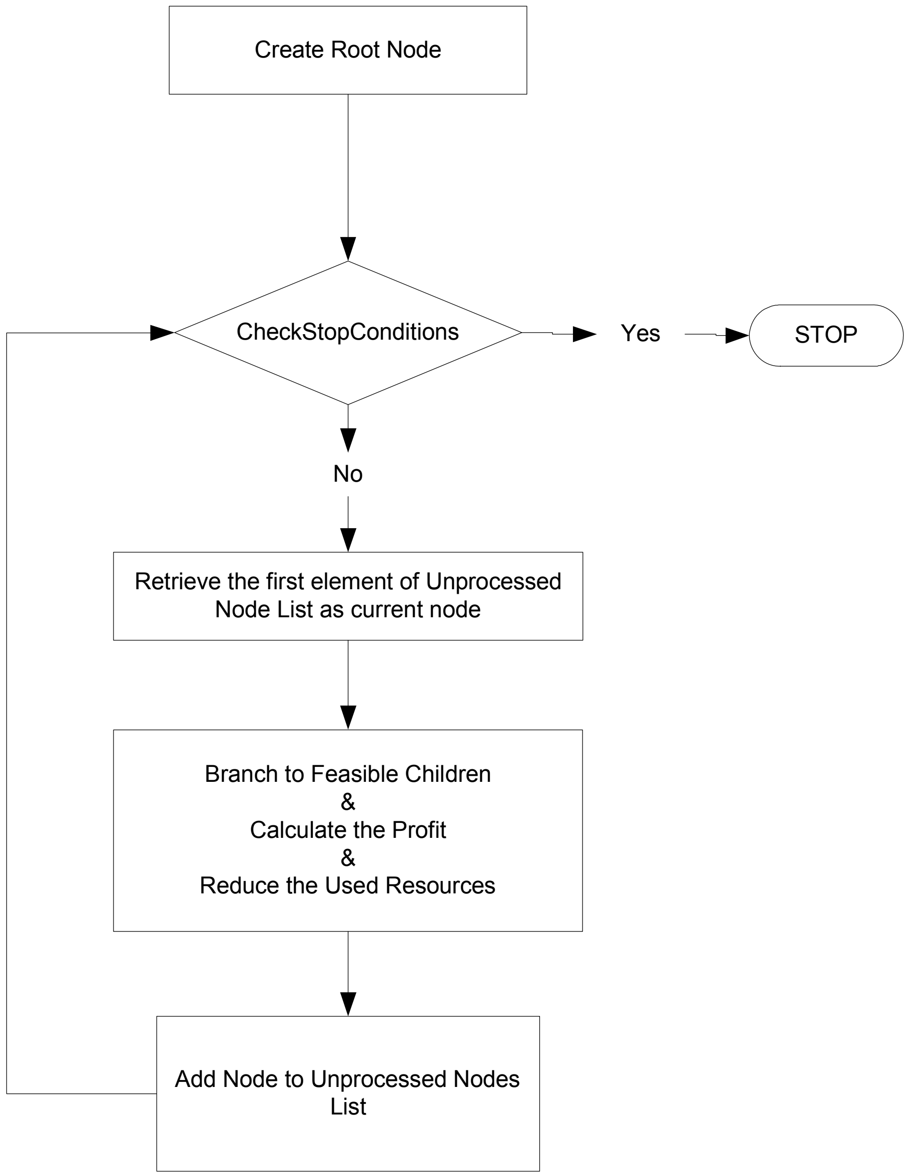


Figure 5.3 - Flow Diagram of BB-GS: Depth-First Search Strategy

This search strategy aims to find feasible solutions firstly. Since the tree is searched vertically, feasible solutions can be obtained in the early phases of the search. Therefore, even the search may be halted due to lack of memory, a feasible solution has already been found. The search starts from the left side of the tree. However, search on the full solution spaces still may not be completed, again because of computational limitations.

5.1.1.3 Best-First Search (BFS)

Different from other two search strategies, Best-First search strategy searches the tree based on a criterion function. In a way, nodes promising to lead to a better solution are selected as the node to be branched. By doing so, it is aimed to reach the solution faster. The flow of the algorithm using Best-First strategy is given below.

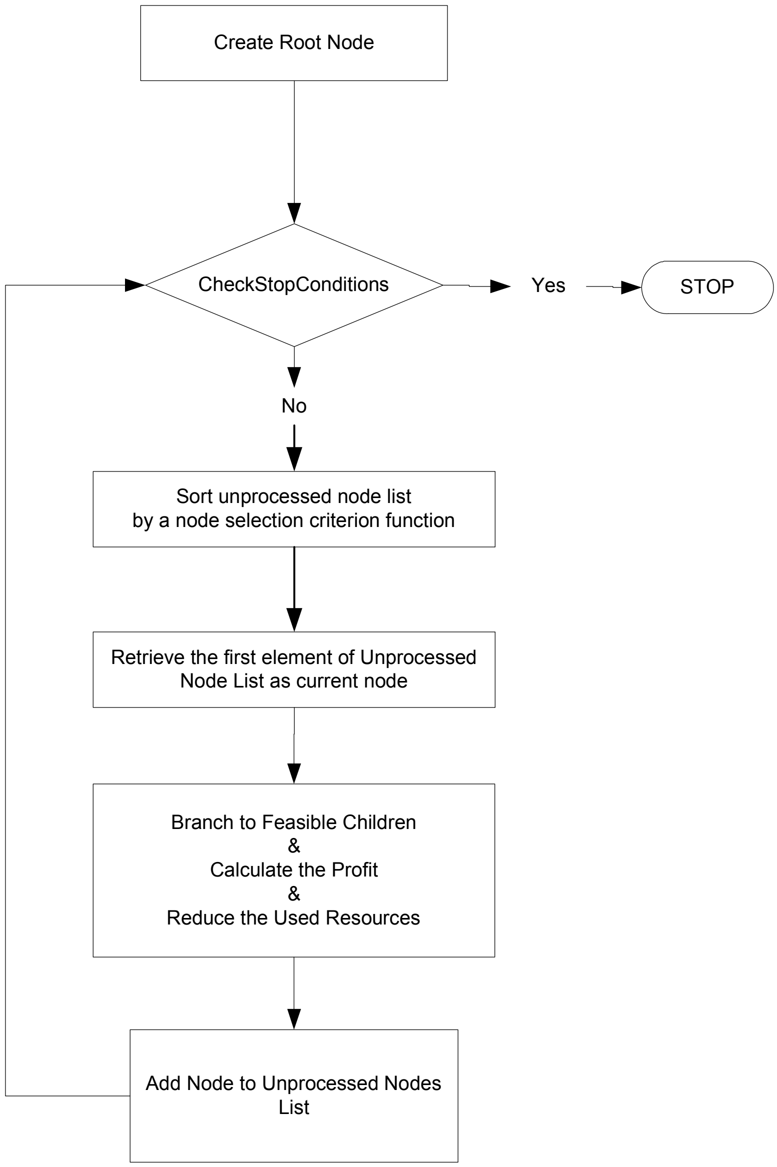


Figure 5.4 - Flow Diagram of BB-GS: Best-First Search

Best-First strategy is somehow similar to Depth-First strategy. In DFS, the first element in the unprocessed node's list is selected as the current node. On the other, in BFS before selection of the current node, unprocessed node list is sorted according to a criterion function. This criterion function shows how desirable a node for selection, by a weighted sum of the present data in a node. In a generic Branch-and-Bound algorithm, bounds can be easily used as the criterion function. Since, in BB-GS, bounds are not calculated and used, data for criterion function is the profit so far the node and the existing resources on the node (numbers of aircrafts in fleets).

5.1.2 Branch-and-Bound – Bounding Case (BB-BC)

In addition to feasibility check, bounding is used for pruning as well. A bound is an estimate on the original problem. It is calculated by ignoring some constraints of the original problem. A more realistic bound calculation will definitely benefits better. On the other hand; it is also desired to use a bounding function which does not require too many CPU time and memory, because this slows down the total search.

It was mentioned in the previous chapter that there is an incumbent value which is the overall best value of the objective so far. Incumbent value refers to a feasible assignment. On the other hand, bound does not necessarily represent a feasible assignment. Since the constraint of the original problem is relaxed, the bound will be most probably higher than the actual optimal value of the original problem. For the problem in hand (maximization problem), if a bound at a node is smaller than the incumbent, it is certain that this node will not lead to the optimum, because even the rough estimate of the optimum value on the node is not higher than the current best found value. The way bounding is used in our study is illustrated below. As it can be observed from the Figure 5.5, calculated bound of node 1 is smaller than the current incumbent. Therefore; the node is fathomed. This node and its successors are pruned.

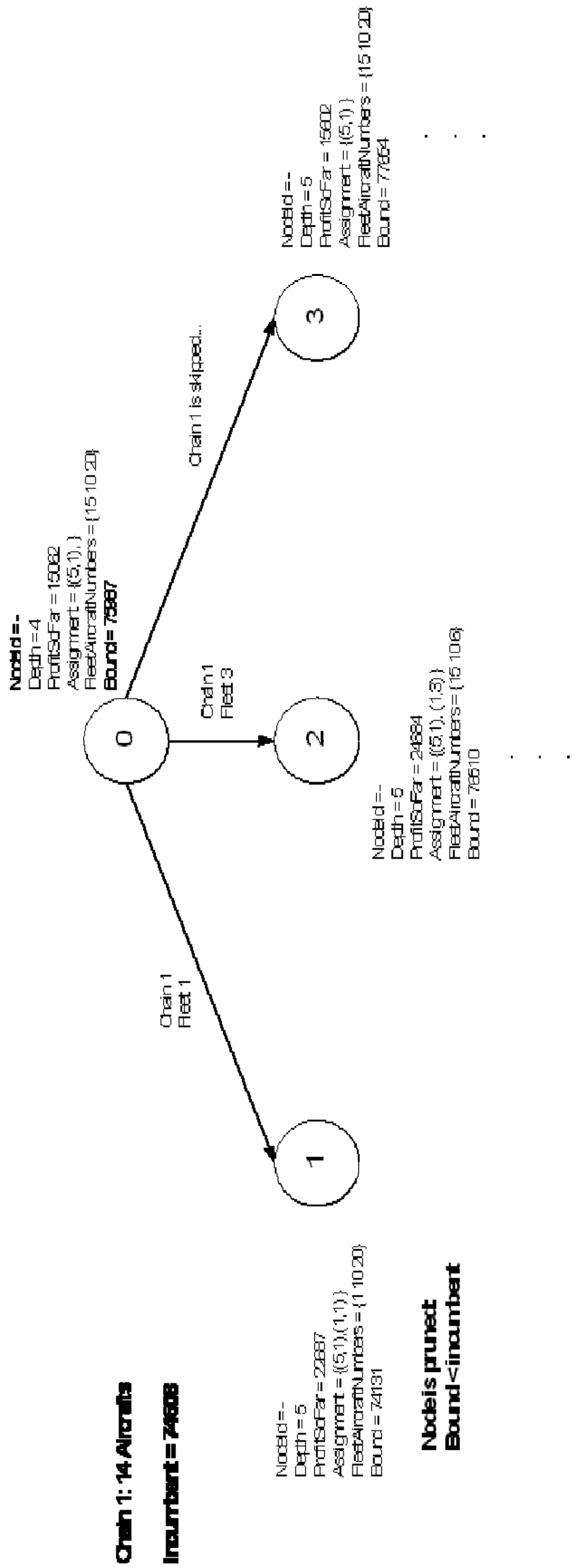


Figure 5.5 - Branch-and-Bound – Bounding Case: Sample Search Tree

5.1.2.1 Bound Calculation & Bounding Function Analysis

As it was mentioned before, on a node, a bounding function is required to return a value which should be as close as possible to the actual solution of the problem that is defined by the node. On the other hand, since the bounding function is run in each node which is created in Branch-and-Bound algorithm, a bounding function with a less computation time is preferable.

Lots of different bounding functions can be defined for the same problem by relaxing different constraints of the original problem. In this section, first, the proposed bounding function is illustrated and examined in details. Latter is about the comparison of the proposed bounding function with a common bound calculation technique; linear programming.

5.1.2.1.1 Proposed Bound Calculation

The constraints of the problem are given in section 3.2. The constraints are grouped under four equations. These constraints are again stated below.

$$\sum_{j=1}^{m} x_{ij} \leq 1 \quad \forall i = 1,\ldots,n \tag{5.1}$$

$$\sum_{i=1}^{n} r_i \cdot x_{ij} \leq b_j \quad \forall j = 1,\ldots,m \tag{5.2}$$

$$\sum_{j=1}^{m} fleetECM_j \cdot x_{ij} \geq chainECM_i \quad \forall i = 1,\ldots,n \tag{5.3}$$

$$\sum_{j=1}^{m} fleetRWR_j \cdot x_{ij} \geq chainRWR_i \quad \forall i = 1,\ldots,n \tag{5.4}$$

For bound calculation, only the constraint given in equation (5.2) is relaxed. This relaxed constraint indicates that total number of aircraft requirements of chains matched with a fleet cannot exceed the number of aircrafts the fleet possesses. The other constraints are somehow used in the bound calculation. ECM and RWR requirement are not relaxed and used in the bounding function. For that purpose, a feasibility matrix has been created which stores feasibility of any possible chain-fleet couple in terms of ECM and RWR requirements and capabilities. If a chain-fleet pair is feasible regarding ECM and RWR compatibility, corresponding element of the feasibility matrix takes value 1, otherwise it takes value 0. Similar to the feasibility matrix, a profit matrix has been created which stores the profit of all possible chain-fleet pairings. A new matrix is defined which is called as "Bounding Function Matrix" and it is calculated by multiplication of feasibility and profit matrices element by element. By this way, in this matrix, infeasible pairs have a zero value. Then, the maximum elements of each row of BFM is taken and stored as BFV. BFV stores the most profitable fleet and profit for each chain, where only the constraint in equation (5.2) is relaxed, others are included. The formulation of the bound calculation function can be illustrated as;

n: number of chains
m: number of fleets

F: Feasibility Matrix	(n x m)
P: Profit Matrix	(n x m)
BFM: Bounding Function Matrix	(n x m)
BFV: Bounding Function Vector	(n x 1)

$$F = \begin{bmatrix} f_{11} & f_{12} & f_{13} & \cdot & \cdot & \cdot & f_{1m} \\ f_{21} & f_{22} & f_{23} & \cdot & \cdot & \cdot & f_{2m} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ f_{n1} & f_{n2} & f_{n3} & \cdot & \cdot & \cdot & f_{nm} \end{bmatrix}$$

$$f_{ij} = \begin{cases} 1 & \text{chain i-fleet j} \rightarrow \text{feasible wrt ECM \& RWR} \\ 0 & \text{chain i-fleet j} \rightarrow \text{not feasible wrt ECM \& RWR} \end{cases}$$

$$P = \begin{bmatrix} p_{11} & p_{12} & p_{13} & \cdot & \cdot & \cdot & p_{1m} \\ p_{21} & p_{22} & p_{23} & \cdot & \cdot & \cdot & p_{2m} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ p_{n1} & p_{n2} & p_{n3} & \cdot & \cdot & \cdot & p_{nm} \end{bmatrix}$$

$$BFM = P. \times F$$

$$BFV = \max\left(BFM^T\right)$$

$$Bound = f\left(chainIndex\right)$$

$$Bound = \sum_{i=chainIndex}^{n} BFV\left(i\right)$$

Algorithm 2 – Calculation of Bounds and Related Variables

5.1.2.1.2 Bounding Function Analysis

The formulation of the problem given in Chapter 3 reveals the fact that the problem is linear. Actually, the type of the problem is referred as 0-1 integer linear programming problem, because the objective function and the constraints are linear and the decision variable is restricted to take 0 or 1. For that particular problem, a linear programming based bounding function can be easily considered.

For a feasible solution of a maximization problem, it has been known that the objective value of a 0-1 integer linear programming problem cannot be more than objective value of the linear programming problem form of the same problem. Since the constraint on the decision variable of 0-1 integer linear programming problem is tighter, the objective value will be more restricted. In the literature, it is shown that a dual problem of a linear program can be defined [34]. The primal form of the program is a minimization problem and the dual problem is a maximization problem. Primal linear program, dual linear program and original 0-1 integer linear program are formulated in their generic form as follows.

$$\begin{array}{lll} \text{minimize} & c^T \cdot x & \\ \text{subject to} & A \cdot x \geq b & \text{(OBJ1)} \quad \text{Primal Linear Program} \\ & x \geq 0 & \end{array} \tag{5.5}$$

$$\begin{array}{lll} \text{maximize} & \lambda^T \cdot b & \\ \text{subject to} & \lambda^T \cdot A \le c^T & \text{(OBJ2)} \quad \text{Dual Linear Program} \\ & \lambda \ge 0 & \end{array} \tag{5.6}$$

$$\begin{array}{lll} \text{maximize} & \lambda^T \cdot b & \\ \text{subject to} & \lambda^T \cdot A \le c^T & \text{(OBJ3)} \quad \text{0-1 Integer Linear Program} \\ & \lambda \in \{0,1\} & \end{array} \tag{5.7}$$

The formulation of our original problem in vector and matrix notation is given in equation (5.7). In equation (5.6), a constraint of the original problem on the bounds of the decision variable is relaxed. This formulation is the dual form of a minimization linear program. The primal form of this dual formulation is given in equation (5.5). The relation of these formulations in terms of the comparison of their objective value is below.

$$\underbrace{\overset{\text{Primal Linear Program}}{\begin{array}{ll} \text{minimize} & c^T \cdot x \\ \text{subject to} & \mathrm{A} \cdot \mathrm{x} \ge \mathrm{b} \\ & \mathrm{x} \ge 0 \end{array}} \ \ge \ \overset{\text{Dual Linear Program}}{\begin{array}{ll} \text{maximize} & \lambda^T \cdot b \\ \text{subject to} & \lambda^T \cdot A \le c^T \\ & \lambda \ge 0 \end{array}}}_{\text{Duality Theorem}} \ \ge \ \overset{\text{Original Problem}}{\begin{array}{ll} \text{maximize} & \lambda^T \cdot b \\ \text{subject to} & \lambda^T \cdot A \le c^T \\ & \lambda \in \{0,1\} \end{array}}$$

Therefore, a feasible solution to the Primal Linear Program can be used as a bound to the original problem. The original problem is converted into the vector form and then the linear programming problem is obtained. The dual linear program form of our problem and its primal form are illustrated as follows.

n: number of chains	
m: number of fleets	
λ: decision variable	nxm X 1
P: profits vector	nxm X 1

V: matrix for constraint given at (3.14)	n X nxm
1: vector whose all elements are 1	n X 1
R: matrix indicating the number of required aircrafts (3.15)	m X nxm
B: vector indicating the aircraft capacities of fleets (3.15)	m X 1
FE: matrix indicating the ECM capacity of fleets (3.16)	n X nxm
FR: matrix indicating the RWR capacity of fleets (3.16)	n X nxm
CE: vector indicating the ECM requirement of chains (3.16)	n X nxm
CR: vector indicating the RWR requirement of chains (3.16)	n X nxm

$$
\begin{aligned}
&\text{maximize} \quad \lambda^T \cdot P \\
&\text{subject to} \\
&\qquad \lambda^T \cdot V^T \leq 1^T \\
&\qquad \lambda^T \cdot R^T \leq B^T \\
&\qquad \lambda^T \cdot \left(-FE^T\right) \leq -CE^T \\
&\qquad \lambda^T \cdot \left(-FR^T\right) \leq -CR^T \\
&\qquad \lambda \geq 0
\end{aligned}
$$

Algorithm 3 – Converting the Formulation to Vector Form

Then, the primal linear program form of the problem can be written as;

$$
\begin{aligned}
&\text{minimize} \quad c^T \cdot x \\
&\text{subject to} \\
&\qquad A \cdot x \geq b \\
&\qquad x \geq 0 \\
&\text{where} \\
&\qquad c^T = 1^T + B^T - CE^T - CR^T \\
&\qquad A = V^T + R^T - FE^T - FR^T \\
&\qquad b = P
\end{aligned}
$$

Algorithm 4 – Primal Linear Program

The bound calculation methods proposed in Algorithm 2 and Algorithm 4 are compared. A scenario involving 8 fleets and 8 chains is randomly generated in

MATLAB environment. Since the accuracy and processing speed of the bounding functions are important, these features are evaluated. The approximate value of the exact solution is around 55000. The optimality gap has been calculated relative to this value.

Table 5-2 Comparison of Bound Calculation Methods

	Processing Time (sec)	Bound	Approx. Optimality Gap (%)
Proposed Bounding Function	1.341×10^{-5}	61435	10.47
LP-Based Bounding Function	0.3742	161540	65.95

According to our analysis, it has been observed that the proposed bounding function calculates the bound faster and more accurate, which is everything required from a bounding function.

5.1.2.2 Breadth-First Strategy

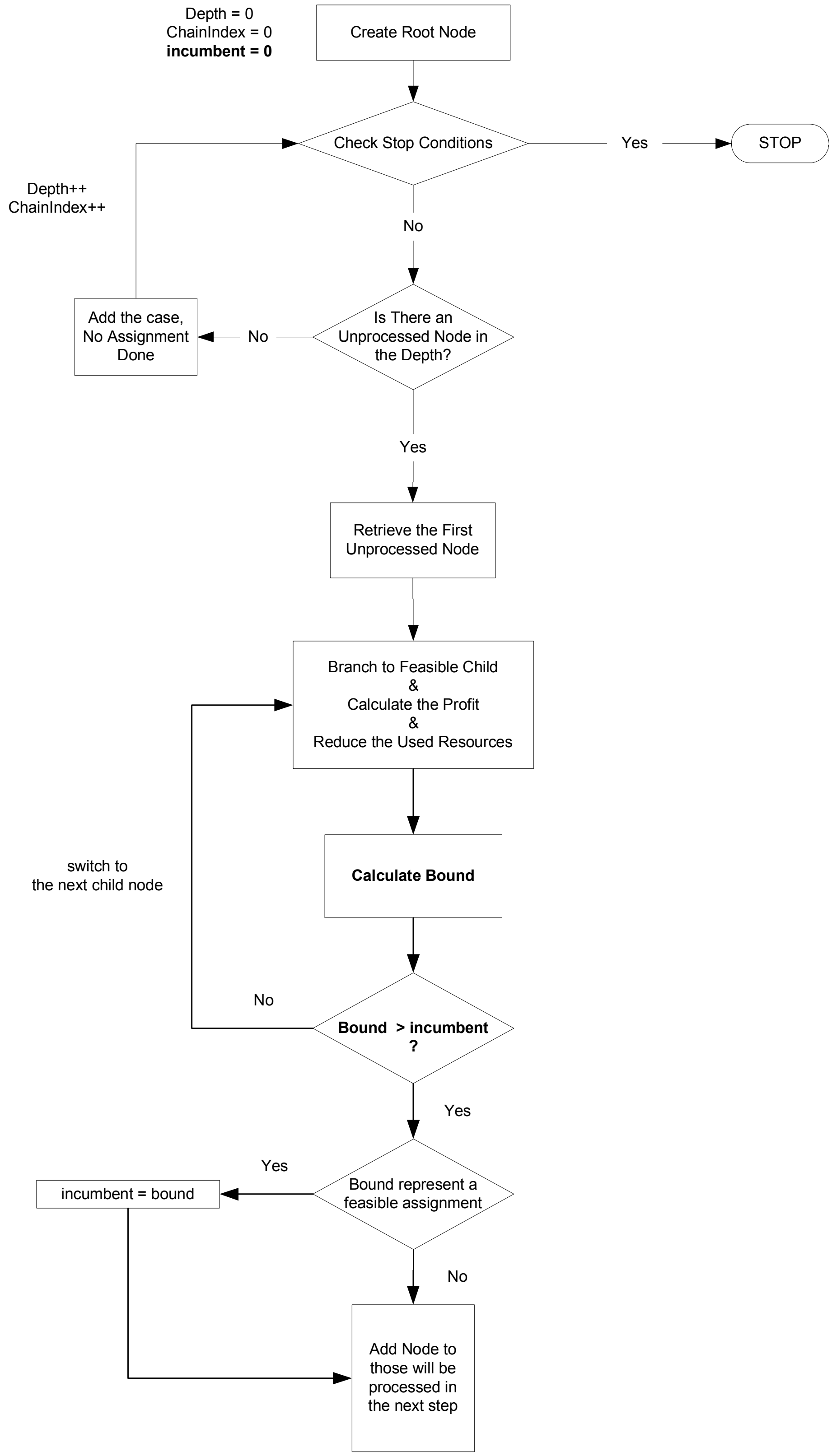


Figure 5.6 - Flow Diagram of BB-BC: Breadth-First Search Strategy

The search strategy is the same. The only difference is that bounding is also used as a pruning method besides feasibility check. The flow of the BB-BC with Breadth-First Search strategy is given in the figure above.

Similar to BB-GS, the feasible children are evaluated, but not included in the search immediately. Their bounds are calculated and compared with the incumbent value. If the bound is lower than the incumbent value, then the node is not qualified as a promising node and excluded from the search. If the bound is greater than the incumbent, the node is included in the search. In cases, where the bound is greater than incumbent and it refers to a feasible assignment, then incumbent is updated with the bound.

5.1.2.3 Depth-First Strategy

Flow diagram is given in Figure 5.7.

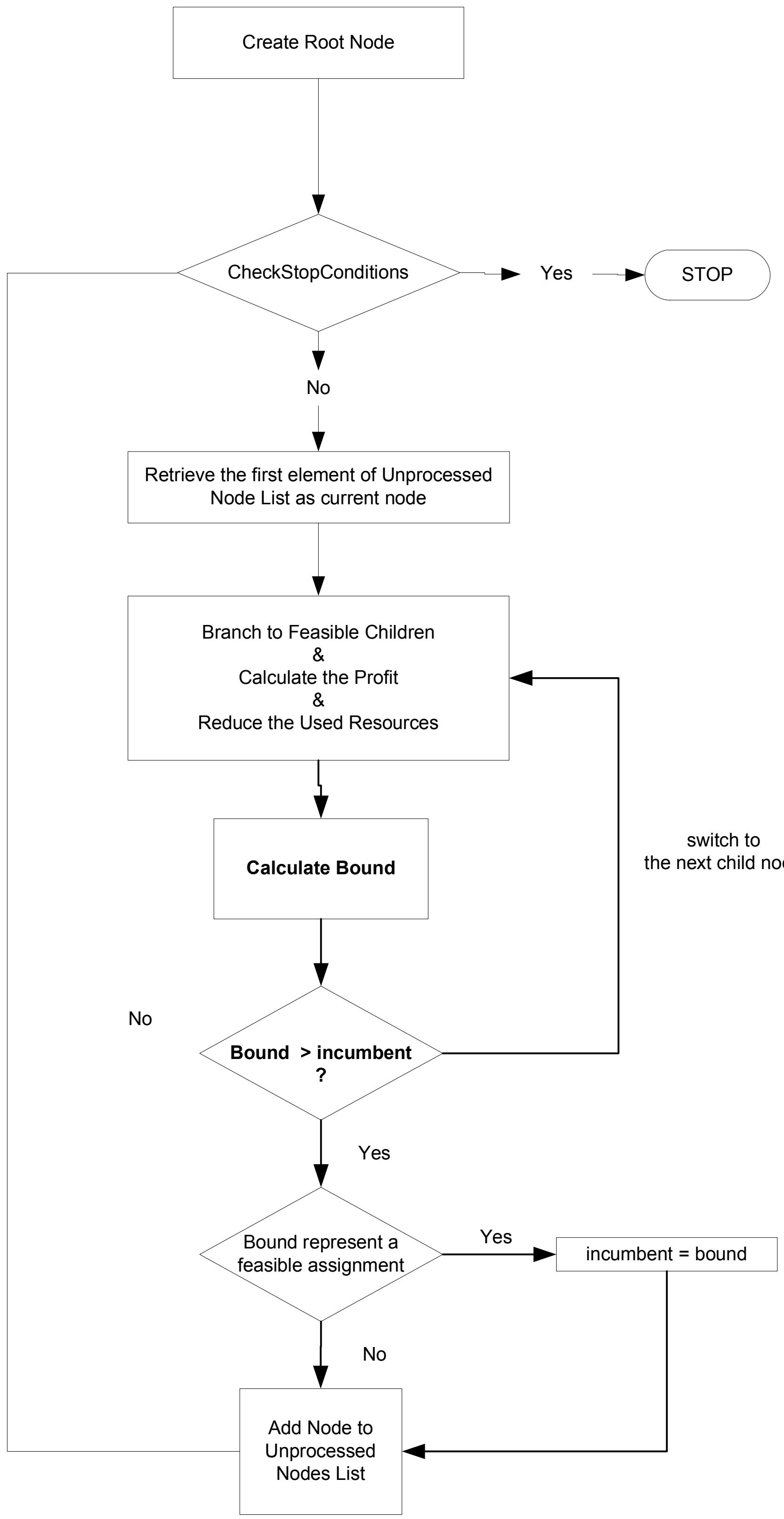


Figure 5.7 - Flow Diagram of BB-BC: Depth-First Search Strategy

5.1.2.4 Best-First Strategy

In Figure 5.7, the first element of the unprocessed node list is selected as the current node. In Best-First search strategy, unprocessed node list is sorted according to a criterion function. The flow of BFS is exactly same with DFS except the criterion function.

Implemented algorithm, Branch-and-Bound – Bounding Case algorithm with Best-First strategy, is very similar to A* search algorithm. A* search algorithm is a graph theory search algorithm which searches for the minimum distance path between two nodes [36]. Similar to Branch-and-Bound algorithm, node selection strategy is performed related to the criterion function calculated by the bound (estimate on the upcoming nodes) and already achieved objective value. However, this particular implemented algorithm is not referred as A* search algorithm. Because different from A* search algorithm, in the implemented algorithm, the whole assignment is not reconstructed by back processing from the final node to initial node. It is preferred to delete all processed nodes for better usage of memory.

5.2 Genetic Algorithm (GA)

Genetic algorithm, with operators and specialties, is explained in Chapter 4. The way, genetic algorithm is applied to the specific problem, is discussed in this section. A special chromosome structure has been used for the specific case study. This special chromosome structure is useful for both operators (crossover, mutation) and also for feasibility analysis of chromosomes. Chromosome structure with the specific gene structure is given in the tables below.

Table 5-3 Gene Structure of Genetic Algorithm

GENE		
Variable	**Data Type**	**Explanation**
Chain Id	short	Unique Id of the Chain
Fleet Id	short	Unique Id of the Fleet
Num. of Used Aircrafts	integer	Number of Aircrafts used in this pairing

Table 5-4 Chromosome Structure of Genetic Algorithm

CHROMOSOME		
Variable	**Data Type**	**Explanation**
Chromosome	vector<Gene>	Unique Id of the Chain Size = nm n: number of chains m: number of fleets
Objective Value	float	Objective value of the assignment
Fitness	float	Fitness of the assignment in the population
Remaining Aircrafts Numbers	Array[short]	Remaining aircraft numbers of fleets from the assignments

In Genetic Algorithm, for an assignment problem, a gene stores only the paired couple. However; we have included the number of aircrafts used in this pairing, because when performing crossover, this number should be calculated in each step. By including it in the gene structure, processing load is minimized.

Besides storing the assignment, the objective value of the assignment is also stored in the chromosome. The objective value is the total profit of the assignment. As it can be remembered from the earlier discussions, the fitness of a chromosome indicates the strength of the chromosome in the population. Although there are many ways for calculating fitness of a chromosome, the selected formula for fitness calculation is given in below.

PS: size of the population

i: index of the ith chromosome in the population

$$fitness_i = \frac{objValue_i}{\sum_{j=1}^{PS} objValue_j}$$

Algorithm 5 – Calculation of the Fitness Value

Number of aircrafts left in the fleets is stored as the parameter “Remaining Aircrafts Numbers”. This variable is totally for feasibility check of the assignment. Genes imposing infeasibility to the chromosome are processed in iteration and they are converted to feasible genes by special feasibility operators. Since the initial population is chosen to be feasible, only factor that can create infeasible chromosomes are the crossover and mutation operators. The parameter, “Remaining Aircrafts Numbers”, is mostly useful at that point.

The general flow of the implemented Genetic algorithm is given below. The flow has some differences from the generic Genetic algorithm flow. Every element of the implemented Genetic algorithm is explored in detail. Random operations are done excessively. A seed is assigned to the random generator for consistency of the results.

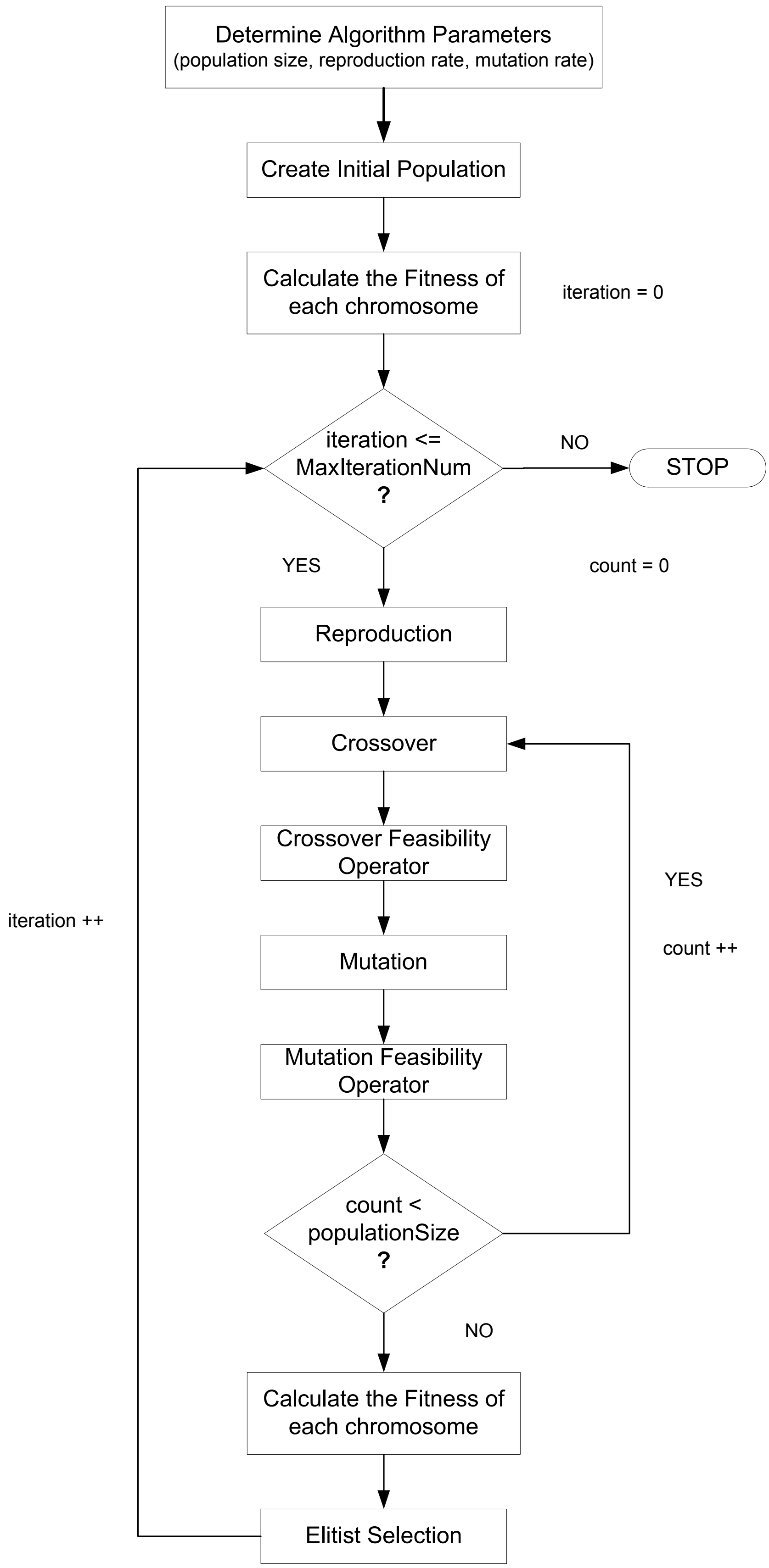


Figure 5.8 - Flow Diagram of the Implemented Genetic Algorithm

5.2.1 Determination of algorithm parameters

Size of the population, reproduction rate, mutation rate and maximum iteration number are important parameters of the algorithm. Although these parameters can be input to the Genetic algorithm class, their certain values should be emphasized. Reproduction rate indicate the number of chromosome which are put to the mating pool for coupling.

$$0 < reproductionRate \leq 1$$

$$0 \leq mutationRate \leq 1$$

5.2.2 Creation of Initial Population

An initial population of feasible assignment is created. This initial population is created by the use of Branch-and-Bound algorithm. Random branching has been performed on the solution tree until a leaf node has been reached. This process has been repeated population size times.

5.2.3 Reproduction

The chromosomes for reproduction are selected randomly from the population. Number of chromosomes selected is population size multiplied by the reproduction rate.

5.2.4 Crossover

Two point and single point crossover techniques are used in this study. The selected chromosomes for reproduction are mated by either of the crossover techniques. The crossover points are randomly chosen. The resulting new chromosomes are then included into the population.

5.2.5 Crossover Feasibility Operator (CFO)

As the crossovers are completed, all the population is scanned for infeasible chromosomes. If CFO finds infeasibility in a chromosome, the chromosome is searched for the genes causing the infeasibility. Those genes are fixed. What CFO does is it finds the most profitable and feasible assignment near the original infeasible assignment. Therefore; CFO does not only cure the infeasibility, but it moves the assignment to the most profitable one around.

5.2.6 Mutation

Mutation is done in a predetermined rate called as mutation rate. As it can be remember from section 5.2.1, mutation rate is between 0 and 1. Each chromosome in the population is checked for mutation. A random number is generated between 0 and 1 for each chromosome. If this random number is greater than mutation rate, then no mutation is done for that particular chromosome at the current iteration. Otherwise, mutation is performed. The gene which is going to be mutated is selected randomly. A random number is generated to specify the gene to be mutated. The fleet of the selected gene is randomly altered. A sample mutated gene is given in the figure below.

Chain Id	**12**	**13**	**8**	**26**	**28**	**39**	**42**	**1**	**23**	**14**	**18**	**51**	**53**	**2**
Fleet Id	**1**	**6**	**8**	**3**	**2**	**4**	**8**	**6**	**5**	**5**	**1**	**8**	**2**	**8**

Chain Id	**12**	**13**	**8**	**26**	**28**	**39**	**42**	**1**	**23**	**14**	**18**	**51**	**53**	**2**
Fleet Id	**1**	**6**	**8**	**3**	**2**	**4**	**8**	**6**	**5**	**5**	**1**	**8**	**2**	**8**

Chain Id	**12**	**13**	**8**	**26**	**28**	**39**	**42**	**1**	**23**	**14**	**18**	**51**	**53**	**2**
Fleet Id	**1**	**6**	**8**	**3**	**2**	**4**	**8**	**2**	**5**	**5**	**1**	**8**	**2**	**8**

Figure 5.9 - Sample Mutation on a Sample Chromosome

As it can be observed from Figure 5.9, gene (1,6) is selected for mutation. Then the fleet Id of this gene is changed to remaining fleets. Moreover, it can be also chosen to assign that selected chain to none of the fleets. Therefore, the list of the selectable fleets, for that example is: {-1, 1, 2, 3, 4, 5, 7, 8}.

5.2.7 Mutation Feasibility Operator (MFO)

Infeasible chromosomes created by the mutation operator are converted to feasible chromosomes be MFO. Mutated gene is controlled whether it imposes infeasibility or not. If mutation causes a mutation, then alternative mutations for the specific gene are considered. The most profitable alternative is selected. Similar to CFO; besides converting an infeasible chromosome to a feasible chromosome, MFO also obtains the best alternative.

5.2.8 Elitist Selection

The overall population with newly generated chromosomes is evaluated in terms of their fitness. Elitist selection has been applied in this study. Fittest elements of the population, whose number is as population size, are selected to be carried onto the new iteration.

5.3 BB-BC with Incumbent Information by Genetic Algorithm (HYBRID I)

In BB-BC as the tree is being generated, the incumbent value is also updated. Since there is no known valid full assignment in the beginning of the search, incumbent is assigned a weak value. Since this weak incumbent value is rarely greater than the bounds, search becomes incapable for pruning nodes via bounding.

Genetic algorithm is run before Branch-and-Bound algorithm in order to overcome this problem. After maximum number of iteration is reached, the objective value of the fittest chromosome of the population is taken as the incumbent value for Branch-and-Bound algorithm. Since Genetic algorithm is run for a particular amount of iteration, the fittest chromosome represents a very good assignment with a very good objective value. As the bounds of nodes are compared with this objective value more pruning can be performed because this incumbent is relatively a better one.

This algorithm is implemented for all tree search strategies. They are referred as;

- HYBRID I – BreFS
- HYBRID I – DFS
- HYBRID I – BFS

5.4 Branch-and-Bound Initiated by GA Partial Assignment (HYBRID II)

As it was for HYBRID I algorithm; in HYBRID II, information from previously run genetic algorithm is used to initiate the Branch-and-Bound algorithm from a better initial state. In this algorithm, assignment of the fittest chromosome is retrieved. A certain portion of this assignment is kept constant and remaining parts are deleted. By this way, a partial assignment is formed. The profit (objective value) of this partial assignment is then calculated. This partial assignment is converted to a node on a tree and this node is used as the root node of the tree. Branch-and-Bound search is initiated from this root node which has a partial assignment within.

The idea behind HYBRID II is to improve the result obtained by Genetic algorithm. Since Branch-and-Bound algorithm searches all the possible matching; starting with the partial assignment root node, it may find a better assignment whose objective value is better greater than the result obtained by GA. Therefore, better results can be obtained without dealing with computational difficulties.

It should be stated that this partial assignment is better to be a long one, because in cases it is not a long assignment, there may still be a huge portion of the solution space for exploration. Therefore, computational limitations can still be a problem.

These processes are exemplified in the figure given below.

Assignment retrieved from GA

Chain Id	**12**	**13**	**8**	**26**	**28**	**39**	**42**	**1**	**23**	**14**	**18**	**51**	**53**	**2**
Fleet Id	**1**	**6**	**8**	**3**	**2**	**4**	**8**	**6**	**5**	**5**	**1**	**8**	**2**	**8**

Selected Partial Assignment

Chain Id	**12**	**13**	**8**	**26**	**28**	**39**	**42**	**1**	**23**	**14**	**18**	**51**	**53**	**2**
Fleet Id	**1**	**6**	**8**	**3**	**2**	**4**	**8**	**6**	**-**	**-**	**-**	**-**	**-**	**-**

assignments to be processed

Figure 5.10 - Retrieving of Partial Assignment & Root Node to Branch-and-Bound

In the example given in Figure 5.10, assignment from chains 12 to 1 are kept constant, which is to say that the tree has been already searched for these chains. This partial assignment is taken to the root node and the searched is performed starting with the chain 23. Moreover, it should be stated that the resources used by the partial assignment is also reduced from the overall resources. Therefore, the root node is shaped with the resources in hand.

This algorithm is implemented for all tree search strategies and for Branch-and-Bound with global search (BB-GS) and Branch-and-Bound with bounding case (BB-BC). These variants are referred as;

- HYBRID II – BB-GS - BreFS
- HYBRID II – BB-GS - DFS
- HYBRID II – BB-GS – BFS
- HYBRID II – BB-BC - BreFS
- HYBRID II – BB-BC - DFS
- HYBRID II – BB-BC - BFS

5.5 Neighborhood Branch-and-Bound initiated by GA (HYBRID III)

This algorithm is again initiated by GA; information retrieved by a priori run of GA is used. The assignment represented by the fittest chromosome is retrieved as the reference assignment. Then, Branch-and-Bound algorithm is started from the root node which has no assignment within. The key point about this algorithm is that as Branch-and-Bound algorithm advances, only the branches within a certain neighborhood around the reference assignment are processed. Therefore, total number of nodes processed lessens and computational load has been reduced. Inevitably, the neighborhood (distance) notion should be introduced first, before advancing to the details of the algorithm.

5.5.1 Neighborhood (Distance)

A distance is defined as the distance between neighboring branches on a tree. Each different element of the assignment vector indicates one unit distance between assignments. Although it has not been detailed until this point, but there is a distance parameter in node definition as it can be remembered from Table 5-1. An example on how distance is used on a Branch-and-Bound search tree is illustrated at Figure 5.11.

Let us assume that allowed distance on tree (distance threshold) is chosen as 2 which means that nodes having a distance to the reference assignment more than 2 are not going to be evaluated. As illustrated in Figure 5.11, starting from the root node on which no assignment exists, the search is performed normally. Children nodes are evaluated as it was done, previously. On the other hand, if the single assignment performed in the children node is not equivalent to the single assignment of the reference assignment at the corresponding depth, then the distance is incremented by one. Children nodes inherit the distance of their parent node. For the example given in the figure, the reference assignment is {b, c, a}. As children nodes of the root node are evaluated, the node represents the single assignment "b" is on the same branch with the reference assignment. Therefore, the distance on this node is said to be zero. On the other hand, for that specific depth, node representing single assignments "a" and “c” are said to have 1 unit distance. Then, in the second depth, the single assignment of the reference assignment is “c”. Nodes having this single assignment do not add additional distance; they only inherit the distance from their parent node. As this process continues, distance on each node is evaluated. In Figure 5.11, red nodes have a 3 unit distance which means that these nodes are not included in the unprocessed node list. Therefore, these nodes are pruned. Green nodes represent valid nodes. Blue nodes are for the reference assignment. As it can be observed from the figure, the distance on each node of the reference assignment branch is zero.

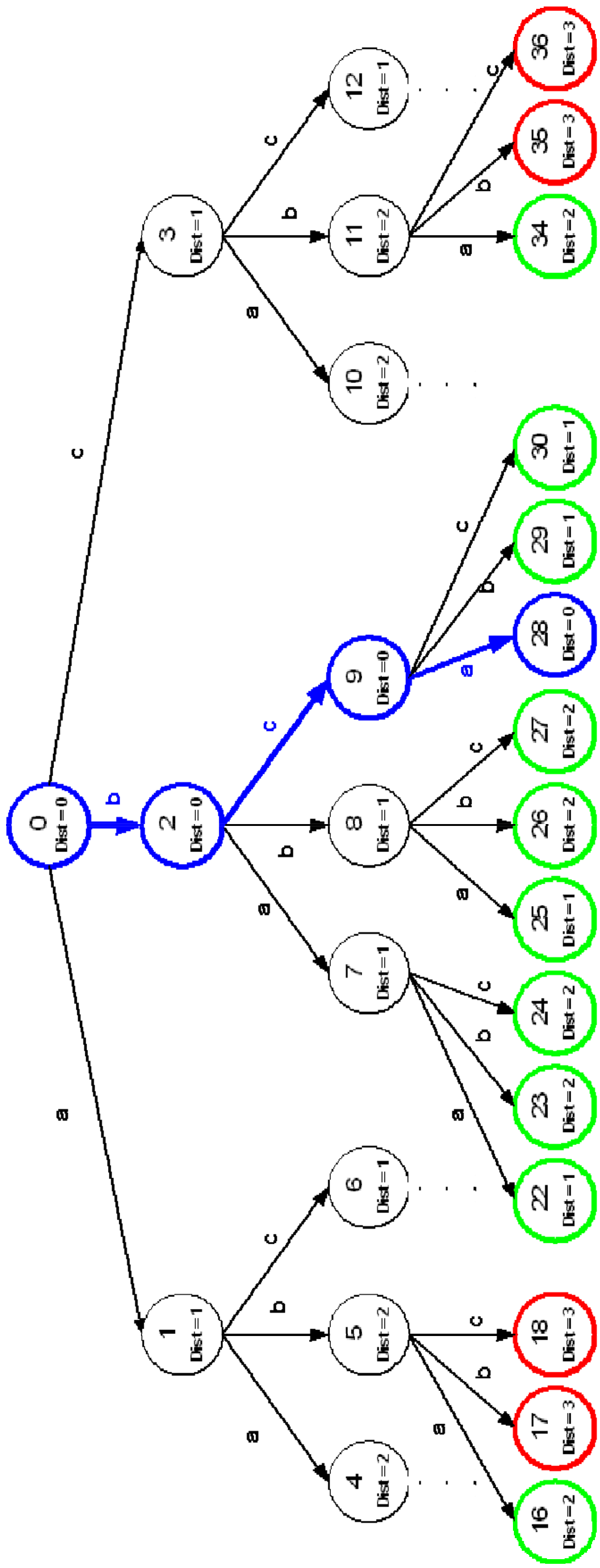


Figure 5.11 - Illustration of Neighborhood on Tree (Distance Concept)

The selection of the distance threshold plays an important role in this algorithm. If the distance threshold is kept high, then the number of total nodes to be processed gets to high because the pruning via distance analysis is low. Computational problems are most likely to be observed. If the distance threshold is kept to low, then the number of evaluated branches and nodes will not be sufficient and improved results may not be found.

5.5.2 Application in the Algorithms

By applying the distance analysis on nodes; on the tree, only the branches having a certain distance to the branch of the reference assignment are evaluated. HYBRID III algorithm which is the application of this distance concept is illustrated in the figure below.

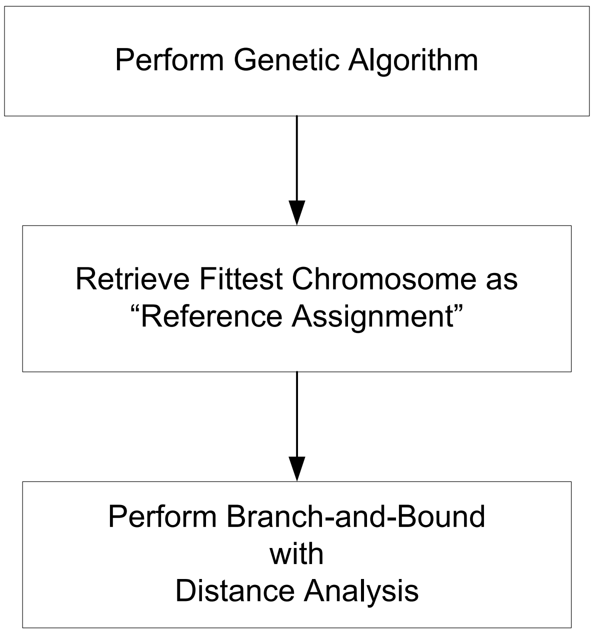


Figure 5.12 - Flow Diagram of the HYBRID II Algorithm

Considering two implemented Branch-and-Bound structure defined with tree search strategies for each, HYBRID III algorithm is classified within itself as follows;

- HYBRID III – BB-GS - BreFS
- HYBRID III – BB-GS - DFS

- HYBRID III – BB-GS – BFS
- HYBRID III – BB-BC - BreFS
- HYBRID III – BB-BC - DFS
- HYBRID III – BB-BC - BFS

CHAPTER 6

SCENARIOS AND RESULTS

The scenarios corresponding to the specific fleet-chain assignment problem are defined. Results are examined for analyzing and evaluating the performance of the implemented algorithms. The algorithms are implemented in C++ programming language for computational considerations. The algorithms are run on a computer equipped with 1 Gb RAM and 1.83 GHz Intel Core 2 processor.

Implemented algorithms are listed below.

- BB-GS: Branch-and-Bound Algorithm with global search in which only the feasibility check has been used for pruning. No bounding function is used. The algorithm is tested for all of three search strategies (BreFS, DFS and BFS)
- BB-BC: Branch-and-Bound Algorithm with bounding function in which bounding is used as a pruning method as well. The algorithm is tested for all of three search strategies (BreFS, DFS and BFS) Initial incumbent value

which is critical for pruning is assigned to zero. It is updated as the algorithm advances.

- Genetic Algorithm: Specific genetic algorithm which is designed for fulfilling the specific needs of the case study problem. Most operations performed in Genetic algorithm are done in random means.
- HYBRID I: A hybrid algorithm using both Genetic and Branch-and-Bound algorithms. The initial incumbent value of Branch-and-Bound algorithm is assigned as the objective value of the fittest chromosome of the population by Genetic algorithm. Branch-and-Bound algorithm is tested for all of three search strategies (BreFS, DFS and BFS)
- HYBRID II: Second hybrid algorithm using both Genetic and Branch-and-Bound algorithms. Certain portion of the best assignment returned from Genetic algorithm has been taken as the partial assignment. Instead of starting from a root node which has no assignment within, Branch-and-Bound algorithm has been initiated from the root node having this partial assignment. Branch-and-Bound algorithm has been tested for all tree search strategies of both BBGS and BBBC.
- HYBRID III: Third hybrid algorithm using both Genetic and Branch-and-Bound algorithms. The best assignment returned form Genetic algorithm has been taken as the reference assignment. Branch-and-Bound (BBGS & BBBC) algorithm is started from a root node where no assignment is involved. Differently, branches which are within a certain neighborhood (distance) of the reference assignment branch are investigated. Previously mentioned search strategies are also used.

A scenario defines all the necessary information for performing the processes and search. Chains, targets within chains and fleets are parametrically expressed in the extent of the scenario. There are four tested scenarios and for each of them all the implemented algorithms mentioned above are tested. Each of these scenarios is for testing different aspects of the algorithms and their behaviors against problems of different sizes and constraint densities. Since the size of the problem is the most important parameter of assignment problems and the performance of the algorithms primarily depends on it, scenarios illustrating problems of different sizes are formed. Moreover, some scenarios are imposing loose constraints and some are tight constraints. For example, all of the chain may require ECM equipped fleets, yet some fleets may not have ECM devices. These cases create more infeasible elements in the solution space.

The results of the algorithms involving Genetic algorithm may differ from run to run. Therefore, a seed is assigned to the random number generator so that the results are always the same as the same seed is supplied. Results of more runs of the algorithms are illustrated for the purpose of testing the efficiencies of the algorithms for different seeds (different initial states).

6.1 Scenario 1: 22 Chains – 8 Fleets

In this scenario, there are 22 chains and 8 fleets. Since one fleet can be matched with more than one chain, 8 fleets can easily be sufficient for testing. Therefore, it does not harm to state that the size of the assignment problem is 22 in which pairings for 22 chains are the scope of the search. The size of this assignment problem can be declared as between a medium-scale assignment problem and a large-scale assignment problem.

The constraints of this scenario are recalled as loose due to a few reasons. Each chain has ECM and RWR requirements and each fleet is sufficient to satisfy these requirements. The numbers of aircrafts required by the chains are moderate; each fleet has enough aircrafts for being able to be matched with any one of the chains.

Considering the size of the problem and the amount of difficulty that the constraints are imposing, only a small portion of the solution space represents infeasible solutions. Therefore; search over a space where only a few solutions are infeasible, is relatively difficult.

The results of the algorithms for this particular scenario are given in the following subsections.

6.1.1 BB-GS

BB-GS algorithm is run for the particular scenario in tree different search strategies which are Breadth-First, Depth-First and Best-First search strategies. The results are given in Table 6-1.

Table 6-1 Results of Scenario 1 for BB-GS Algorithm

	# Nodes	Profit	Processing Time (sec)	# Unassigned Chains
BreFS	11976156	49260.9	150	14
DFS	4913379	75839.8	90	0
BFS	2459895	76339.2	600	0

As one of the stop conditions of BB-GS, a processing time limit is determined. In processing time column, a red color indicates that the algorithm is halted due to memory requirement. This time is the one just before memory of the machine is full.

Regarding the results of Table 6-1, it can be concluded that none of the search strategies has coped to navigate through the full search tree. BreFS has been halted while processing depth 8. Therefore, BreFS has failed to return a full assignment and the profit out of this search strategy is very low relative to other two search strategies. The advantage of the DFS and BFS search strategies is obvious through the results. Although the global optimum cannot be returned, these two strategies are managed to obtain full assignments. The profit obtained by BFS is higher than the profit obtained by DFS. This is expected since BFS uses a criterion function which leads the algorithm to the potential nodes (assignment) primarily. On the other hand, the processing time of BFS is higher than the other two search strategies. This result is an expected one as well, since the list of the unprocessed nodes is sorted according to the criterion function and this sorting operation becomes extremely expensive when the size of the list is relatively high.

For this particular scenario, we can conclude that BB-GS is not efficient because it cannot search the whole search tree due to memory requirements. The number of created nodes is high which indicate low pruning. Application of feasibility check as the only metric for pruning does not return efficiency, which is also expected, since the constraints of the scenario are loose on the fleets.

6.1.2 BB-BC

The results of the particular scenario for BB-BC are illustrated in the table below. The results are given for tree search strategies.

Table 6-2 Results of Scenario 1 for BB-BC Algorithm

	# Nodes	Profit	Processing Time (sec)	# Unassigned Chains
BreFS	13092465	49260.9	150	14
DFS	7388365	75928.8	90	0
BFS	1428678	76371.7	600	0

Similar to BB-GS; BB-BC has failed to search to whole search tree as well which is again expected considering the size of the problem. With the help of the bounds on nodes, better results are obtained. In the same algorithm processing time limits, BB-BC has obtained more profitable assignments compare to BB-GS. The reason is; in BB-BC, the nodes having no probability of leading to better results are pruned. Therefore, algorithm searches more potential nodes during the same processing time and finds better solutions.

6.1.3 Genetic Algorithm

Since the results of Genetic algorithm depends on the random numbers which are effective in creating the initial population, reproduction, crossover and mutation phases. Therefore, in order to analyze the performance of Genetic algorithm, a few numbers of seeds are assigned to the random number generator and multiple runs have been done. 20 runs are made and the seeds are 1….20. Selected parameters and the plots of this multiple runs are as follows.

Population Size:	100
Max. Number of Iterations:	250
Reproduction Rate:	0.5
Mutation Rate	0.3

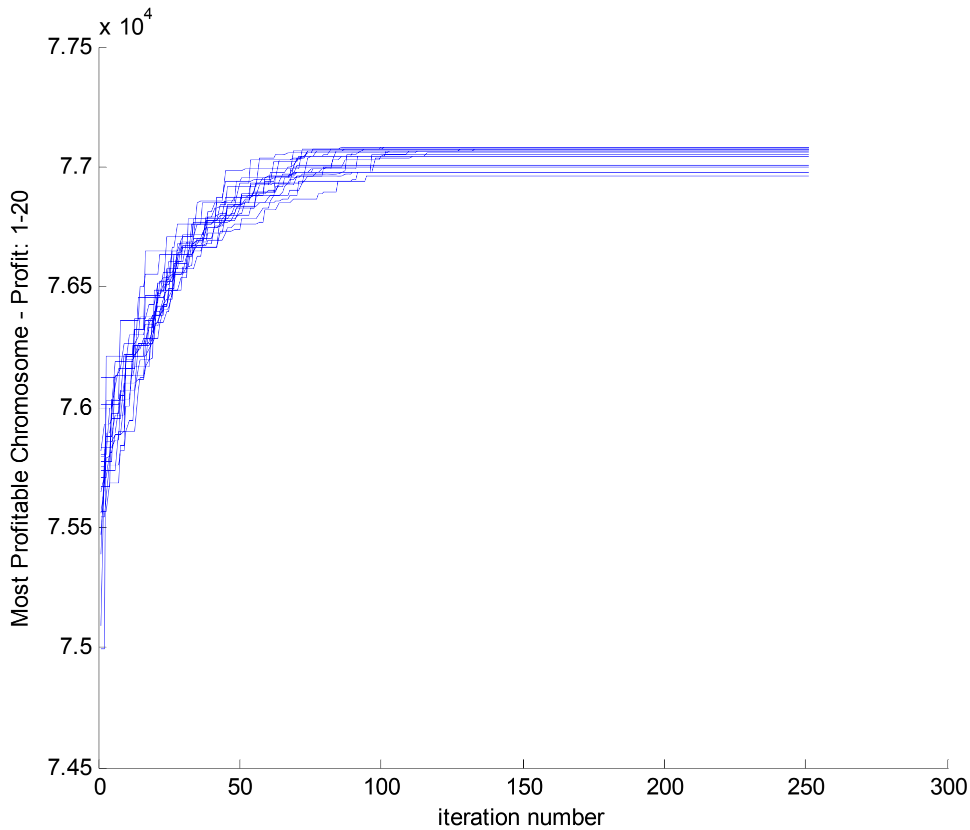


Figure 6.1 - Results of Scenario 1 for Genetic Algorithm (20 runs)

Although the characteristic of each single run cannot be discriminated from Figure 6.1, the general characteristic of the Genetic algorithm for the particular scenario is definitely visible. As the iteration number advances, the algorithm finds more profitable assignments. After a certain number of iteration is done, then the population settles where improvements cannot be found any more. Among 20 runs, the one with the maximum final profit and minimum final profit are plotted in Figure 6.2. The green line represents the maximum profit, red line represents the minimum profit and the dashed blue line gives the mean of 20 runs. Data out of 20 runs are given below.

Mean:	77051	
Maximum Profit:	77080	(seed = 1)
Minimum Profit:	76953	(seed = 2)

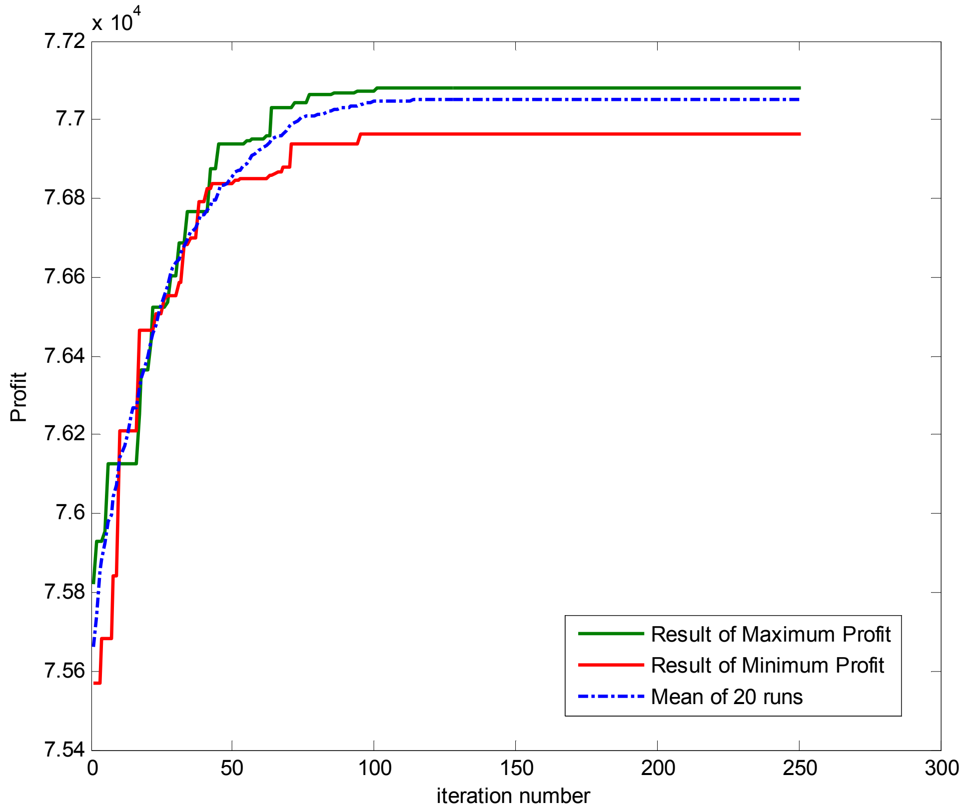


Figure 6.2 - Maximum, Minimum and Mean Results of GA for Scenario 1

Considering the results given above, Genetic algorithm seems to be a very effective algorithm for the particular scenario. Regarding the basic known algorithms, genetic algorithm is inevitably advantageous compare to Branch-and-Bound algorithm.

One of the most important observations that should capture our attention is that, Genetic algorithm has reached its steady state point at an early iteration. Implemented Genetic algorithm forces infeasible assignments to the most profitable feasible assignments. This property makes the algorithm eager to obtain better results and they are found in the early phases of the genetic algorithm.

6.1.4 Hybrid I

The key concept of Branch-and-Bound algorithm is the use of bounds for pruning of nodes which are not leading to the optimal. For maximization problems, if bound calculated on a node is lesser than the incumbent value, then the node is pruned. Therefore, a better incumbent will definitely improve the performance of the algorithm. In Hybrid I algorithm, the result of the Genetic algorithm is used as the initial incumbent value for Branch-and-Bound algorithm. The minimum profit and maximum profit which are found by Genetic algorithm (6.1.3) are used and tree search strategies are tested. Results are as below.

Table 6-3 Results of Scenario 1 for Hybrid I Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	76953	52739.4	9158023	300	13
	DFS	76953	76062.4	2821815	120	0
	BFS	76953	76371.7	1652632	600	0
Maximum Profit by GA	**BreFS**	77080	55803.8	8512447	300	13
	DFS	77080	73877	2213582	120	0
	BFS	77080	76371.7	1653178	600	0

As the results of the Hybrid I algorithm are analyzed, it can be concluded that for the maximization problem, better incumbent decreases the size of the search tree which is in interest. Compare to BB-BC, a better initial incumbent has definitely improved the results of Branch-and-Bound algorithm. For minimum profit by GA, fewer nodes are processed and yet better results are obtained. Via a better incumbent value, pruning rate is increased. In BB-BC, BreFS created too much nodes and only 14 chains are left unassigned. By the use of Hybrid I, 13 chains are left unassigned and a more profitable assignment is found. Moreover, DFS and BFS are also more efficient; they have found more profitable results while they explore less number of nodes. Since creating fewer nodes will avoid causing memory difficulties, Hybrid I algorithm is definitely more preferable than BB-GS and BB-BC. On the other hand, for this particular scenario, Hybrid I has failed to improve the results obtained by GA.

6.1.5 Hybrid II

As it was mentioned before, Hybrid II uses partial assignment retrieved from the fittest chromosome of GA. Then, Branch-and-Bound algorithm is initiated from this partial assignment and results are expected to be improved. For this scenario, the size of the partial assignment is selected as 17. Since the search is initiated from this partial assignment, the Branch-and-Bound search investigates the assignments for the remaining 5 chains. The results for Hybrid II are given below. In Table 6-4, BB-GS is used as the Branch-and-Bound algorithm. In Table 6-5, BB-BC is used as the Branch-and-Bound algorithm.

Table 6-4 Results of Scenario 1 for Hybrid II - BBGS Algorithm

		Profit (GA)	**Profit**	**# Nodes**	**Processing Time (sec)**	**# Unassigned Chains**
Minimum Profit by GA	**BreFS**	76953	76953	5184	<1	0
	DFS	76953	76953	2121	<1	0
	BFS	76953	76953	2121	<1	0
Maximum Profit by GA	**BreFS**	77080	77080	3903	<1	0
	DFS	77080	77080	1861	<1	0
	BFS	77080	77080	1861	<1	0

Table 6-5 Results of Scenario 1 for Hybrid II - BBBC Algorithm

		Profit (GA)	**Profit**	**# Nodes**	**Processing Time (sec)**	**# Unassigned Chains**
Minimum Profit by GA	**BreFS**	76953	76953	6	<1	0
	DFS	76953	76953	110	<1	0
	BFS	76953	76953	110	<1	0
Maximum Profit by GA	**BreFS**	77080	77080	1191	<1	0
	DFS	77080	77080	1191	<1	0
	BFS	77080	77080	1191	<1	0

Regarding the results, it is obvious that Hybrid II has done no good in improving the results for the particular scenario. The reason can be explained as follows. The assignment returned by Genetic algorithm is sorted according to the value of the chains assigned. Therefore, the most valuable assignments are in the beginning of the assignment vector. Since the partial assignment is taken, the retrieved partial assignment stores the most valuable single assignments. Hybrid II does no improvement, since the single assignments that can improve the result are kept constant in the partial assignment.

6.1.6 Hybrid III

By using the result of the Genetic algorithm as a reference assignment, Hybrid III searches the solution tree in a way that only the branches within a certain neighborhood (distance) to the branch of the reference assignment are searched. Since the size of the problem is large and it is desired to avoid from computational difficulties, distance is selected as 3. The results are illustrated below.

Table 6-6 Results of Scenario 1 for Hybrid III - BBGS Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	76953	77065.5	803349	28	0
	DFS	76953	77065.5	803349	28	0
	BFS	76953	77065.5	803349	239	0
Maximum Profit by GA	**BreFS**	77080	77080	791388	27	0
	DFS	77080	77080	791388	27	0
	BFS	77080	77080	735096	240	0

Table 6-7 Results of Scenario 1 for Hybrid III - BBBC Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	76953	77065.5	365836	17	0
	DFS	76953	77065.5	365836	16	0
	BFS	76953	77065.5	365836	85	0
Maximum Profit by GA	**BreFS**	77080	77080	356935	16	0
	DFS	77080	77080	356935	16	0
	BFS	77080	77080	356935	91	0

For the particular scenario, Hybrid III seems to be the only heuristic which has been capable of improving the results. As it can be observed from Table 6-6 and Table 6-7, the concept of neighborhood search on a tree is successful considering the fact that profit calculated by GA (76953) has been elevated to 77065.5. Hybrid III with BB-BC is more successful, because it has improved the results by creating fewer nodes on the tree. Therefore, the distance can be increased more until memory requirements are not exceeded and results can be further improved.

6.1.7 On Results of Scenario 1

Since the size of the problem which is defined by this specific scenario is large, it has been previously foreseen that Branch-and-Bound algorithm is incapable in its any form. The results obtained by Genetic algorithm are very satisfactory. Especially implemented heuristics for performing feasible and profitable crossover and mutation operations have proven their efficiency regarding the fact that the search for optimal is aggressive and agile. Tree proposed algorithms which are for improving the results are tested whether they are fulfilling their mission of creation. Hybrid I and Hybrid II algorithms fail to be successful and make no improvement on the results of Genetic algorithm. Meanwhile, Hybrid III has been observed as an efficient way for improving the results, since it has been successful for such a difficult scenario where the constraints are loose and problem size is large.

6.2 Scenario 2: 22 Chains – 8 Fleets

In Scenario 2, constraints on the fleets are tight. Each chain in the scenario requires both ECM and RWR capability, yet 3 of 8 fleets are not equipped with necessary devices. Therefore, the scenario represents a problem whose size is actually 22 chains - 5 fleets. The reason for such a scenario is to observe whether the performance of the algorithms will change depending on the scenario constraints.

The results of the algorithms for this particular scenario are given in the following subsections.

6.2.1 BB-GS

BB-GS algorithm is run for the particular scenario in tree different search strategies which are Breadth-First, Depth-First and Best-First search strategies. The results are given in Table 6-8.

Table 6-8 Results of Scenario 2 for BB-GS Algorithm

	# Nodes	Profit	Processing Time (sec)	# Unassigned Chains
BreFS	13633075	55558.4	150	12
DFS	13917211	68990.9	90	7
BFS	5607453	69208.8	600	7

In BB-GS, all three of the search strategies have failed to search the whole solution space. Since the constraints are tight, all chains cannot be assigned to a fleet where there is no fleet left having aircrafts to be used. Again, DFS and BFS have returned better results, although the optimal has not found.

For this particular scenario, we can conclude that BB-GS is not efficient because it cannot search the whole search tree due to memory requirements. The number of created nodes is high which indicate low pruning. The behavior of the BB-GS in this scenario is very similar to the one for the Scenario 1.

6.2.2 BB-BC

Table 6-9 Results of Scenario 2 for BB-BC Algorithm

	# Nodes	**Profit**	**Processing Time (sec)**	**# Unassigned Chains**
BreFS	13197467	56054.9	150	12
DFS	7221985	69371.4	90	7
BFS	4009998	69368.3	600	7

Similar to BB-GS; BB-BC has failed to search to whole search tree as well which is again expected considering the size of the problem. With the help of the bounds on nodes, better results are obtained. In the same algorithm processing time limits, BB-BC has obtained more profitable assignments compare to BB-GS. Since use of the bounds for pruning decreases the number of created nodes, fewer nodes are created in BB-BC compare to BB-GS. Therefore; at the same amount of processing time, BB-BC has returned better results relative to BB-GS.

6.2.3 Genetic Algorithm

As it was done for Scenario 1, multiple runs have been done for being able to analyze the performance of the algorithm. The plots of this multiple runs are given below.

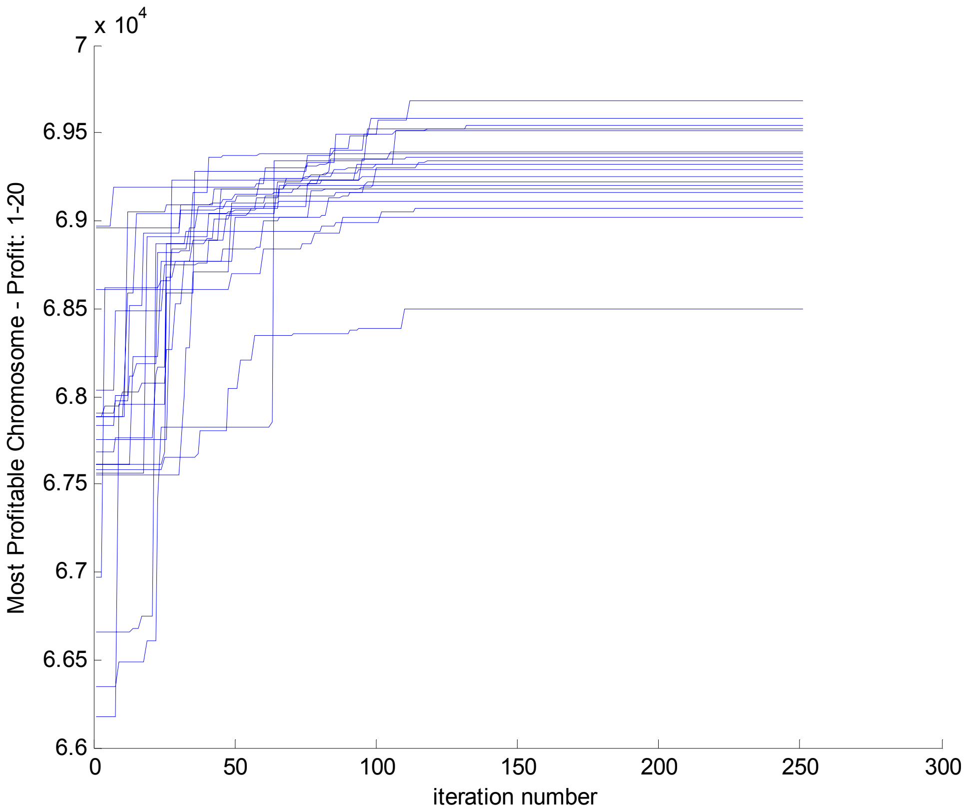


Figure 6.3 - Results of Scenario 2 for Genetic Algorithm (20 runs)

In Scenario 1, the characteristic of Genetic algorithm is quite obvious where results of the runs are very close to each other. On the other hand, results are relatively more separated from each other for Scenario 2. The reason for such a situation is; due to infeasibilities emerging from the tight constraints, lots of created chromosomes are infeasible and feasible crossover and mutation operators as well have difficulties in finding a feasible assignment. Therefore, the population cannot evolve that eagerly and stays around its initial positions.

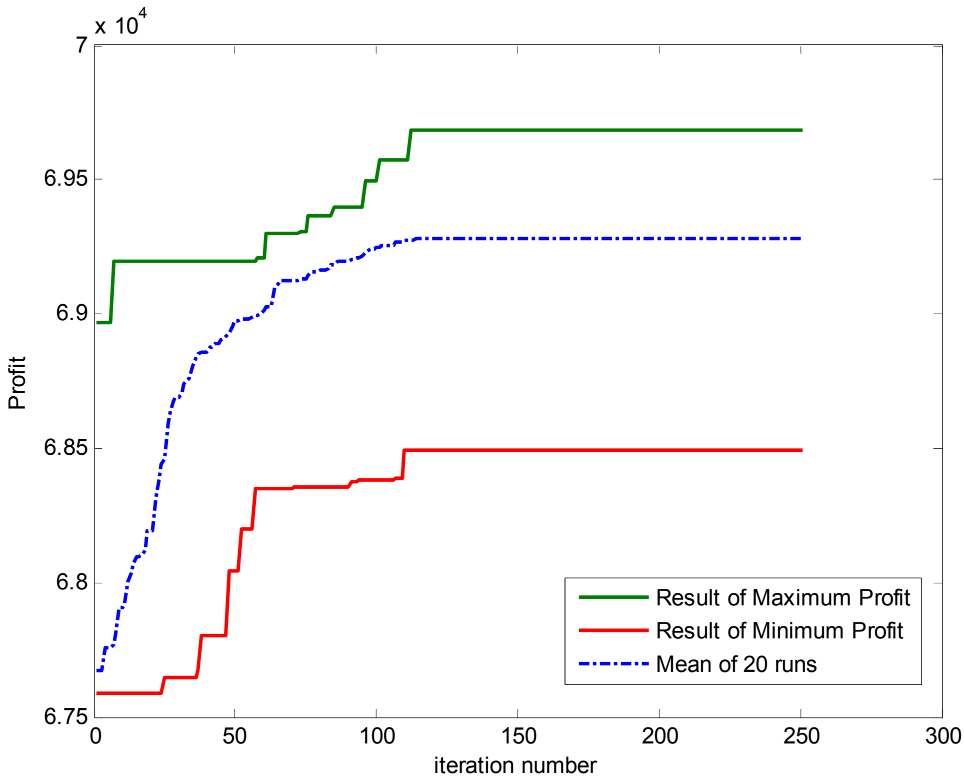


Figure 6.4 - Maximum, Minimum and Mean Results of GA for Scenario 2

The performance of genetic algorithm is again better than Branch-and-Bound algorithm in terms of speed and providing more profitable assignments. Out of 20 runs, the information on assignment with the maximum profit and the assignment with the minimum profit are selected to be carried on to the hybrid methods.

Mean:	69282	
Maximum Profit:	69686	(seed = 15)
Minimum Profit:	68493	(seed = 3)

6.2.4 Hybrid I

Obtained minimum and maximum profits are used in Hybrid I algorithm as the initial incumbent values for Branch-and-Bound algorithm, in our case BB-BC. Corresponding results are given below.

Table 6-10 Results of Scenario 2 for Hybrid I Algorithm

		Profit (GA)	**Profit**	**# Nodes**	**Processing Time (sec)**	**# Unassigned Chains**
Minimum Profit by GA	**BreFS**	68493	59220.2	11313477	300	11
	DFS	68493	69395.8	12705327	120	7
	BFS	68493	69368.3	4020830	600	7
Maximum Profit by GA	**BreFS**	69686	62152	14714077	300	13
	DFS	69686	69395.8	12350917	120	0
	BFS	69686	69368.3	4036272	600	0

Regarding the results above, it can be stated that initial incumbent value enables to obtain more profitable assignments compare to profits obtained for BB-GS and BB-BC. Although it will not be fair to compare the pruning rate of Hybrid I and BB-BC because their processing times are different, it is expected that Hybrid I should prune more. This seems as a fault on the result obtained above.

6.2.5 Hybrid II

The size of the partial assignment which is retrieved from Genetic algorithm and used as the root node assignment for Branch-and-Bound algorithm is selected as 10. The search is started from this root node and possible assignments for the remaining chains are explored. The results of Hybrid I for the particular scenario are illustrated below.

Table 6-11 Results of Scenario 2 for Hybrid II - BBGS Algorithm

		Profit (GA)	**Profit**	**# Nodes**	**Processing Time (sec)**	**# Unassigned Chains**
Minimum Profit by GA	**BreFS**	68493	69245.5	126876	1	7
	DFS	68493	69245.5	102660	1	7
	BFS	68493	69245.5	102660	3	7
Maximum Profit by GA	**BreFS**	69686	69686	223793	2	7
	DFS	69686	69686	177619	1	7
	BFS	69686	69686	177619	6	7

Table 6-12 Results of Scenario 2 for Hybrid II - BBBC Algorithm

		Profit (GA)	**Profit**	**# Nodes**	**Processing Time (sec)**	**# Unassigned Chains**
Minimum Profit by GA	**BreFS**	68493	69245.5	2094	<1	7
	DFS	68493	69245.5	2439	<1	7
	BFS	68493	69245.5	2439	<1	7
Maximum Profit by GA	**BreFS**	69686	69686	1857	<1	7
	DFS	69686	69686	1923	<1	7
	BFS	69686	69686	1923	<1	7

As it can be observed from the tables above; Hybrid II algorithm is successful in terms of improving the results of Genetic algorithm for the partial assignment of the minimum profit run has been used. Meanwhile, the improvement has been achieved in a very short amount of time. Moreover, the importance of bounds in pruning can be observed from the results above regarding the fact that difference between the numbers of created nodes in BB-GS and BB-BC. If BB-GS is used as a part of Hybrid II, then the number of created nodes to obtain the results is around one hundred thousand. On the other hand, for BB-BC, this number is around couple of thousands. Use of bounds provides an increase in pruning rate, so the number of nodes explored lessens.

An important topic that should be discussed is the performance of Hybrid II for Scenario 1 and Scenario 2. The results of Hybrid II algorithm for Scenario 1 reveal the fact that Hybrid II algorithm is not suitable for scenarios like Scenario 1 where all chain can be assigned to a fleet. On the other hand, Hybrid II has improved the results of Genetic algorithm for Scenario 2 which is a modified version of Scenario 1; constraints on fleets are tightened. Due the constraints on the fleets, only 5 of 8 fleets are suitable for assignment. Therefore, there are many chains are left unassigned in returned assignment by GA. Since Hybrid II searches all these unassigned chains out of the partial assignment, it evaluates alternative permutations in which using the resources of a fleet in assignment of multiple chains might be more profitable than using them as the result of Genetic algorithm used.

6.2.6 Hybrid III

The reference assignment retrieved from Genetic algorithm is used for defining the reference branch on Branch-and-Bound search tree. Branch-and-Bound algorithm searches the solution space where only the branches within a certain neighborhood of the reference assignment are evaluated. For Scenario 2, the distance is selected as 4. The results of Hybrid III for the specific scenario are illustrated in Table 6-13 and Table 6-14.

Table 6-13 Results of Scenario 2 for Hybrid III - BBGS Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	68493	69409.1	714367	12	7
	DFS	68493	69409.1	714367	12	7
	BFS	68493	69409.1	714367	95	7
Maximum	**BreFS**	69686	69820.4	603906	9	7

Profit by GA	**DFS**	69686	69820.4	603906	9	7
	BFS	69686	69820.4	603906	85	7

Table 6-14 Results of Scenario 2 for Hybrid III - BBBC Algorithm

		Profit (GA)	**Profit**	**# Nodes**	**Processing Time (sec)**	**# Unassigned Chains**
Minimum Profit by GA	**BreFS**	68493	69409.1	31001	1	7
	DFS	68493	69409.1	136507	3	7
	BFS	68493	69409.1	136491	15	7
Maximum Profit by GA	**BreFS**	69686	69638.8	31070	1	7
	DFS	69686	69820.4	130935	3	7
	BFS	69686	69820.4	130890	14	7

Hybrid III algorithm is successful for this scenario as well. For both of the reference assignments of GA (least profitable and most profitable chromosomes of Genetic algorithm run), Hybrid III has achieved to obtain a more profitable assignment. In Hybrid III – BB-BC, bounds helps to find the solution by exploring fewer nodes.

6.2.7 On Results of Scenario 2

Scenario 2 is very similar to Scenario 1; the only difference is that constraints on the fleets are tighter. Since the constraints are tighter, assigning all of the chains cannot be succeeded. The algorithms are tested for this particular scenario and results are gathered. Branch-and-Bound algorithm is not successful which is normal again considering the size of the problem. Genetic algorithm obtains relatively better assignments, yet the population cannot evolve as desired due to the high density of infeasible assignments in solution space. Among the tree methods proposed to improve the results; Hybrid I has a better performance than BB-GS and BB-BC, but still cannot manage to improve the results further as much as desired.

It has been observed that Hybrid II can be used for this particular type of problem in which assignment on all of the chains cannot be done. As it was in the results of Scenario 1, Hybrid III proves itself as the most convenient method for the problem for Scenario 2 as well.

6.3 Scenario 3: 9 Chains – 8 Fleets

Scenario 3 has a special purpose; the problem defined by this scenario is a small-scale assignment problem and its size is 9. Therefore, for this particular scenario, Branch-and-Bound algorithm should return the global optimum without having any computational difficulties. The results for this specific scenario are illustrated in the following subsections.

6.3.1 BB-GS

BB-GS algorithm is run for the particular scenario in tree different search strategies which are Breadth-First, Depth-First and Best-First search strategies. The results are given in Table 6-15.

Table 6-15 Results of Scenario 3 for BB-GS Algorithm

	# Nodes	Profit	Processing Time (sec)	# Unassigned Chains
BreFS	6059813	56351.9	35	0
DFS	7104071	56351.9	131	0
BFS	7104071	56351.9	454	0

The algorithm has searched the entire solution space and tree search strategies have found the same assignment as the optimal assignment. On the other hand, it should be stated that there has been a faulty measure in calculating the number of created nodes for BreFS.

6.3.2 BB-BC

Results of the BB-BC for the specific scenario are given below.

Table 6-16 Results of Scenario 3 for BB-BC Algorithm

	# Nodes	Profit	Processing Time (sec)	# Unassigned Chains
BreFS	24976	56351.9	1	0
DFS	211846	56351.9	2	0
BFS	1416708	56351.9	118	0

As expected, all tree search strategies have obtained the same assignment which is the one also obtained in BB-GS. Moreover, with the helps of the bounding procedure, the solution has been obtained in less time by exploring fewer amounts of nodes.

It should be stated that the number of created nodes in BreFS should be higher than the numbers for DFS and BFS since DFS and BFS are more probable to update the incumbent value which helps to obtain the solution by creating fewer nodes. At this point, it occurs that the calculation of the number of created nodes is faulty.

6.3.3 Genetic Algorithm

As it was mentioned before, this scenario defines a small-scale problem whose solution can be found by exact solution methods. On the other hand, the efficiency of Genetic algorithm is another topic since it is not guaranteed in GA to find the global optimum. 20 runs are performed whose results are illustrated in the figure below.

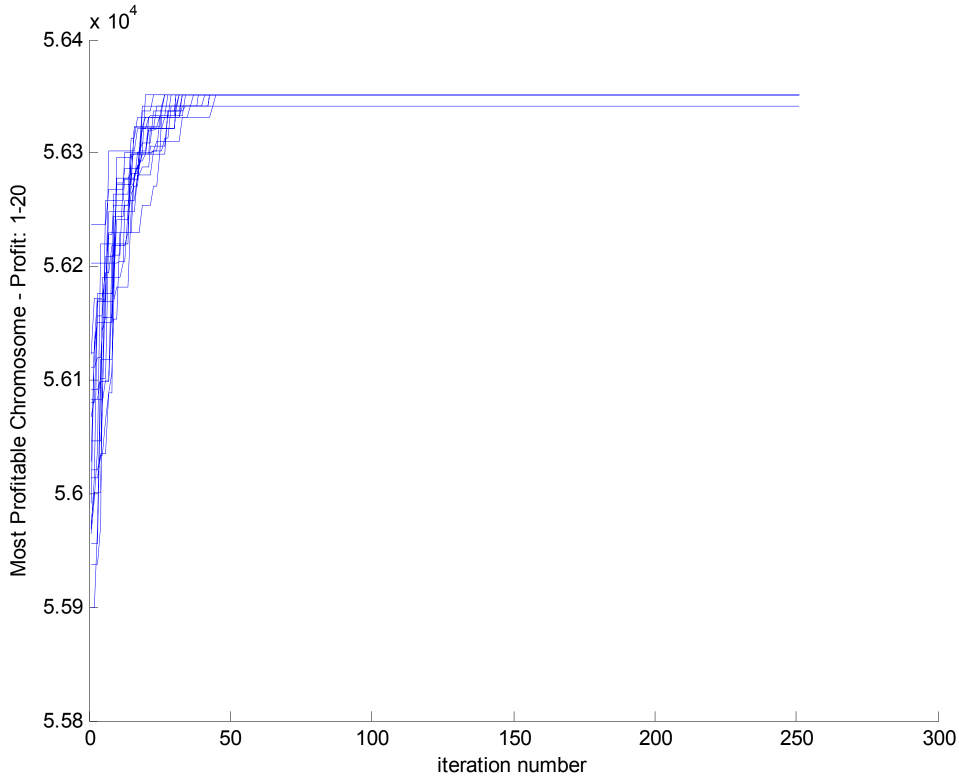


Figure 6.5 - Results of Scenario 3 for Genetic Algorithm (20 runs)

The plot reveals the fact that the result of almost every run is the same. In most of the runs, it does not matter where the initial positions of chromosomes, the same assignment has been achieved. The important results of the overall 20 runs are illustrated as;

Mean:	56351	
Maximum Profit:	56351.9	(seed = 2)
Minimum Profit:	56341	(seed = 1)

Genetic algorithm has also achieved to obtain the global optimal assignment. On the other hand, for some certain runs, global optimum may not be found due to randomness of the operation.

The characteristics of the population of runs returning the maximum and minimum profits are plotted in Figure 6.6.

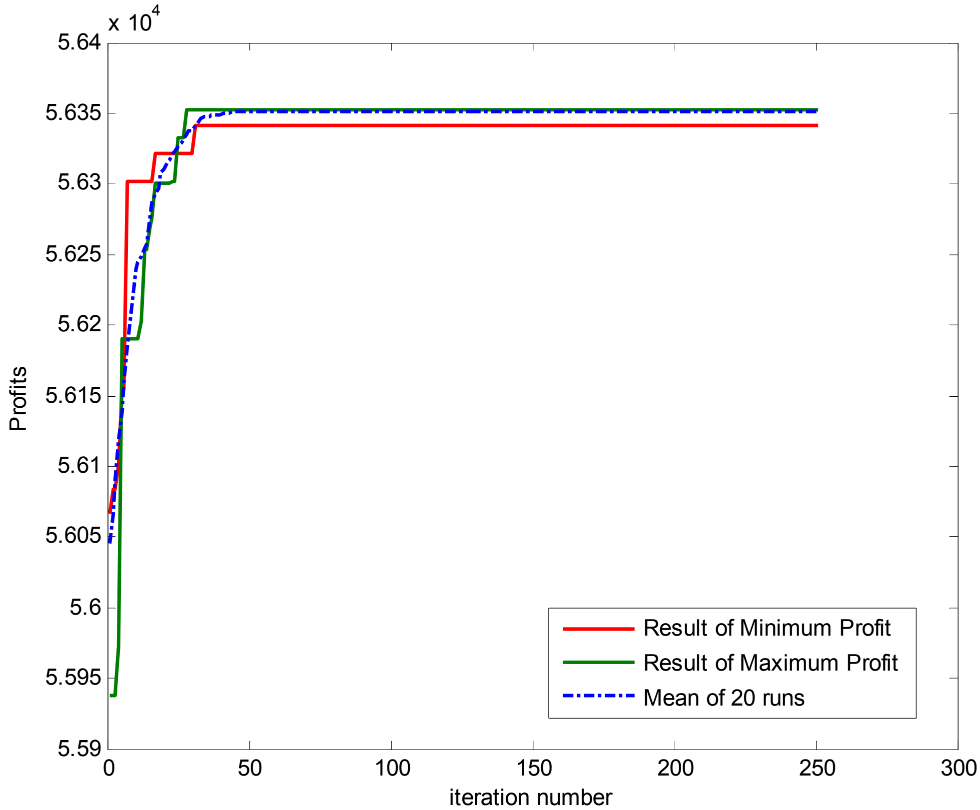


Figure 6.6 - Maximum, Minimum and Mean Results of GA for Scenario 3

6.3.4 Hybrid I

It is obvious that any of the hybrid algorithms can improve the partial assignment of GA's maximum profit, because this partial assignment is already the globally optimal assignment. The minimum and maximum profits calculated by GA are used as the initial incumbent in BB-BC algorithm. The results are as follows.

Table 6-17 Results of Scenario 3 for Hybrid I Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	56341	56351.9	2890	0	0
	DFS	56341	56351.9	211820	2	0
	BFS	56341	56351.9	1391468	108	0
Maximum Profit by GA	**BreFS**	56351.9	56351.9	2476	0	13
	DFS	56351.9	56351.9	211820	2	0
	BFS	56351.9	56351.9	1391468	108	0

Hybrid I algorithm has succeeded to improve the result of the Genetic algorithm run of minimum profit. For these initial incumbent values and search strategies, the global optimum is always obtained.

The effect of a good initial incumbent can be easily observed when the numbers of created nodes in BB-BC and Hybrid I are compared. Since a good initial incumbent value helps to increase the pruning rate, the number of nodes created in Hybrid I is so much less than the number of created nodes in BB-BC.

6.3.5 Hybrid II

The length of the partial assignment is chosen to be 4. The aim is to analyze the performance of Hybrid II for a small-scale problem. The results are given in Table 6-18 and Table 6-19.

Hybrid II is insufficient for improving the results. The reason is the same one of the Scenario 1. Hybrid II algorithm cannot improve the results for scenarios in which there are enough resources to assign all chains. In Scenario 1 and Scenario 3, a full assignment can be obtained and for these scenarios Hybrid II fails to improve the results of Genetic algorithm. On the other hand, in Scenario 2, a full assignment cannot be obtained due to resource constraints and Hybrid II has improved the results.

Table 6-18 Results of Scenario 3 for Hybrid II - BBGS Algorithm

		Profit (GA)	**Profit**	**# Nodes**	**Processing Time (sec)**	**# Unassigned Chains**
Minimum Profit by GA	**BreFS**	56341	56341	4510	<1	0
	DFS	56341	56341	1533	<1	0
	BFS	56341	56341	1533	<1	0
Maximum Profit by GA	**BreFS**	56351.9	56351.9	4731	<1	0
	DFS	56351.9	56351.9	1571	<1	0
	BFS	56351.9	56351.9	1571	<1	0

Table 6-19 Results of Scenario 3 for Hybrid II - BBBC Algorithm

		Profit (GA)	**Profit**	**# Nodes**	**Processing Time (sec)**	**# Unassigned Chains**
Minimum Profit by GA	**BreFS**	56341	56341	1064	<1	0
	DFS	56341	56341	1064	<1	0
	BFS	56341	56341	1064	<1	0
Maximum Profit by GA	**BreFS**	56351.9	56351.9	1150	<1	0
	DFS	56351.9	56351.9	1150	<1	0
	BFS	56351.9	56351.9	1150	<1	0

6.3.6 Hybrid III

Since the size of the problem is 9 and it can be solely solved by Branch-and-Bound, choosing a high distance threshold value will not harm the algorithm in terms of computational limits. The distance threshold is selected as 5. The results are given below.

Table 6-20 Results of Scenario 3 for Hybrid III - BBGS Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	56341	56351.9	541635	9	0
	DFS	56341	56351.9	541635	9	0
	BFS	56341	56351.9	541635	61	0
Maximum Profit by GA	**BreFS**	56351.9	56351.9	540644	9	0
	DFS	56351.9	56351.9	540644	9	0
	BFS	56351.9	56351.9	540644	61	0

Table 6-21 Results of Scenario 3 for Hybrid III - BBBC Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	56341	56351.9	106847	2	0
	DFS	56341	56351.9	106847	2	0
	BFS	56341	56351.9	106847	10	0
Maximum Profit by GA	**BreFS**	56351.9	56351.9	106351	2	0
	DFS	56351.9	56351.9	106351	2	0
	BFS	56351.9	56351.9	106351	10	0

Hybrid III algorithm has succeeded to improve the result of the Genetic algorithm run of minimum profit.

6.3.7 On Results of Scenario 3

Scenario 3 is a crucial scenario, because it is a small-scale assignment problem which should be solely solved by Branch-and-Bound algorithm and global optimal assignment should be found in all cause.

As it is expected, Branch-and-Bound algorithm has succeeded to find the global optimum. The use of bounds has decrease the processing time for the solution. Although it does not theoretically required from Genetic algorithm to find the global optimum, Genetic algorithm has obtained it as well. For cases in which Genetic algorithm does not return the global optimum, Hybrid I and Hybrid III algorithms achieve to carry the result to the global optimum. On the other hand, Hybrid II fails to improve the results. This strengthens our claim that Hybrid II algorithm is for improving the results of scenarios in which there is no enough resources to match all the chains, not for scenarios in which all chains can be matched.

6.4 Scenario 4: 32 Chains – 8 Fleets

Scenario 4 defines a large-scale assignment problem in which there are 32 chains and 8 fleets, yet resources of fleets are not sufficient to match all chains. With this scenario, all possible scenario combinations will be covered. The results for this specific scenario are illustrated in the following subsections.

6.4.1 BB-GS

The results of BB-GS for Scenario 4 are illustrated in the table below.

Table 6-22 Results of Scenario 4 for BB-GS Algorithm

	# Nodes	Profit	Processing Time (sec)	# Unassigned Chains
BreFS	18600700	77655	301	23
DFS	14392107	84660.4	122	19
BFS	8557114	84660.4	600	19

The size of the problem is huge considering the fact that it is an assignment problem. Therefore, Branch-and-Bound algorithm has failed to search the whole solution space. None of the search strategies obtain the optimal assignment. Since there are not enough resources to match all chains, a full assignment cannot be found. Although DFS and BFS attempt to search full assignment first, their profit are not close to the optimal and they can be improved further. BreFS cannot search beyond the half depth of the tree, because of the computational limitations. As it can be observed from the table, in tree search strategies, numbers of created nodes are huge as expected.

6.4.2 BB-BC

The results of BB-BC for Scenario 4 are illustrated in Table 6-23.

Table 6-23 Results of Scenario 3 for BB-BC Algorithm

	# Nodes	Profit	Processing Time (sec)	# Unassigned Chains
BreFS	19537613	79690.8	301	23
DFS	16054474	86526	120	18
BFS	6494244	86374.6	600	18

Although use of bounds increases the profits found by search strategies, the numbers of created nodes are still high which is normal regarding the problem size. Therefore, BB-BC is not suitable for such big problem.

6.4.3 Genetic Algorithm

Considering the size of the problem, Genetic algorithm seems as the most suitable predefined method for this scenario. As it was done for the previous scenarios, 20 runs are performed for Genetic algorithm. The general characteristic of 20 runs can be observed from the following figure.

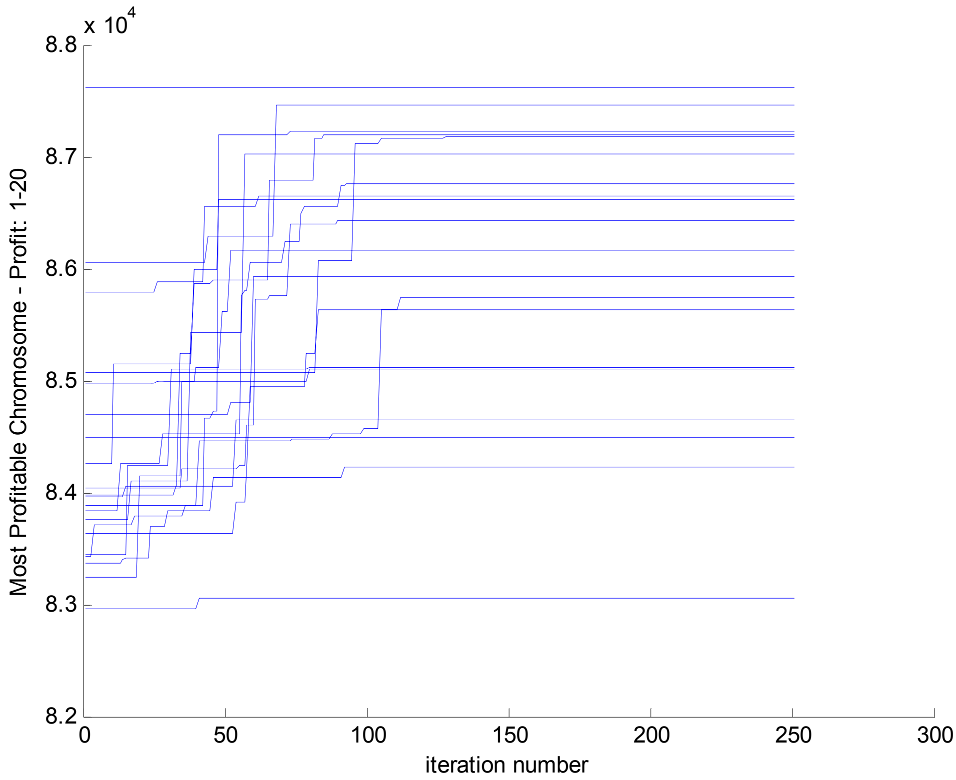


Figure 6.7 - Results of Scenario 4 for Genetic Algorithm (20 runs)

An important topic about this scenario is that some chains of the scenario require relatively large numbers of aircrafts. Therefore, we can state that the constraints of the scenario are tight. Therefore, there are many assignment vectors in the solution space which are infeasible. This condition is forcing Genetic algorithm especially, since randomly generated chromosomes become infeasible so often that population cannot evolve as desired and initial population stays unchanged even after certain amount of iteration has advanced.

Mean:	86013	
Maximum Profit:	87611	(seed = 8)
Minimum Profit:	83056	(seed = 17)

There is huge difference between the maximum profit and minimum profit of GA runs. On the other hand, mean upgrades itself as iteration advances.

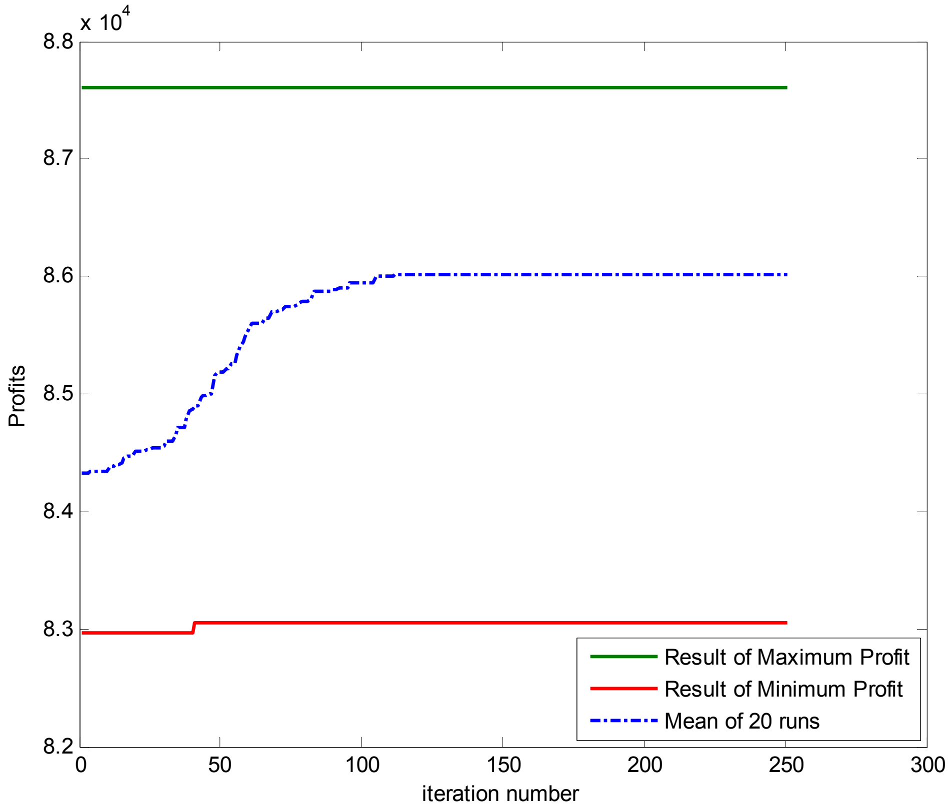


Figure 6.8 - Maximum, Minimum and Mean Results of GA for Scenario 4

As it can be observed from Figure 6.8, populations of minimum and maximum profit runs cannot evolve to a better and fitter population. The initial populations are kept carrying on to the end of iterations. However, Genetic algorithm is still successful in solving the problem and finding acceptable good assignments.

6.4.4 Hybrid I

The maximum and minimum profits calculated by Genetic algorithm are taken to be used as the initial incumbent values for BB-BC. The results of Hybrid I algorithm for Scenario 4 are illustrated in Table 6-24.

The assignments calculated by Hybrid I, are more profitable compare to the ones of Genetic algorithm, BB-GS and BB-BC and improves the results, except BreFS.

Since BreFS searches the solution space horizontally, it cannot get into the deep of the tree and find better assignments. On the other hand, the use of better incumbents definitely improves the performance of Branch-and-Bound algorithm.

Table 6-24 Results of Scenario 4 for Hybrid I Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	83056	79690.8	19806522	300	23
	DFS	83056	86526	16013000	120	18
	BFS	83056	86374.6	6555652	600	18
Maximum Profit by GA	**BreFS**	87611	79690.8	18657787	300	23
	DFS	87611	86383.6	16214960	120	18
	BFS	87611	86222.4	6661719	600	18

6.4.5 Hybrid II

The length of the partial assignment is selected as 9. The results are given below.

Table 6-25 Results of Scenario 4 for Hybrid II - BBGS Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	83056	85511.4	15593	0	20
	DFS	83056	85511.4	11253	1	20
	BFS	83056	85511.4	9951	0	20
Maximum Profit by GA	**BreFS**	87611	88118.9	14909766	300	17
	DFS	87611	88064.9	9823484	120	19
	BFS	87611	88118.9	12293983	600	17

Table 6-26 Results of Scenario 4 for Hybrid II - BBBC Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	83056	85511.4	1885	<1	20
	DFS	83056	83415.9	1179	<1	21
	BFS	83056	83415.9	1179	<1	21
Maximum Profit by GA	**BreFS**	87611	88118.9	101899	2	17
	DFS	87611	86987.4	80331	1	18
	BFS	87611	86987.4	80271	3	18

The general trend of Hybrid II algorithm in improving the results for this particular scenario is positive. Although exceptions are observed, the profit retrieved by GA is improved by different search strategies of BB-GS and BB-BC. For that particular scenario, BreFS has seemed to be the best choice considering there is no computational difficulty encountered.

Since the results above are not building any pattern, we avoid making any judgment on the performance of the method for the specific scenario. On the other hand, it becomes inevitable to mention that the performance of Hybrid II algorithm is better for such scenarios in which all chains cannot be matched with fleets due to resource constraints (tight constraints).

6.4.6 Hybrid III

The results of Hybrid III for Scenario 4 are illustrated in the upcoming tables. The selected distance is 4. The reference assignments of the returned from the runs of Genetic algorithm having the minimum and maximum profit are retrieved and the search tree is searched where neighbor branches are evaluated.

Table 6-27 Results of Scenario 4 for Hybrid III - BBGS Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	83056	87492	4437501	55	20
	DFS	83056	87492	4437501	56	20
	BFS	83056	87492	4437501	418	20
Maximum Profit by GA	**BreFS**	87611	88550.4	4240889	59	18
	DFS	87611	88550.4	4240889	57	18
	BFS	87611	88550.4	4240889	413	18

Table 6-28 Results of Scenario 4 for Hybrid III - BBBC Algorithm

		Profit (GA)	Profit	# Nodes	Processing Time (sec)	# Unassigned Chains
Minimum Profit by GA	**BreFS**	83056	87492	1489097	22	20
	DFS	83056	87492	1489097	22	20
	BFS	83056	87492	1489097	129	20
Maximum Profit by GA	**BreFS**	87611	88550.4	1450595	23	18
	DFS	87611	88550.4	1450595	23	18
	BFS	87611	88550.4	1450595	124	18

Hybrid III algorithm has succeeded to improve the results for both maximum and minimum runs of GA. Once again, it has been the algorithm which finds the maximum profit for the particular scenarios.

6.4.7 On Results of Scenario 4

Scenario 4 is the most complicated scenario, because the size of the problem it is representing is large and the scenario contains chains which are requiring relative large numbers of aircrafts. The constraints of the problems are tight.

Branch-and-Bound algorithm is not successful for such a scenario, which is normal considering the size of the problem. Genetic algorithm has achieved to obtain good results. However; since the number of infeasible solutions is high, population cannot evolve as desired. Hybrid I and Hybrid II algorithms manage to attain more profitable assignments. Hybrid III algorithm arises as the most convenient method, because it improves the results more than any other method and without having any computational problems.

CHAPTER 7

CONCLUSION AND COMMENTS

7.1 On the Study and Results

In this thesis, exploring the overall optimal assignment of air force sources to tasks in an operation has been discussed and the search for attaining the solution maximizing the profit has been done. The problem is referred as an assignment problem. Instead of assigning aircrafts to targets, pairings between chains and fleets are searched. Chain is defined as a group of, at most five, targets which are in a relationship in terms of time precedence, unity and importance in the operation. The optimal assignment is the one maximizing the profit of the operation and yet feasible regarding the resources of the fleets and requirements of the chains. It has been discussed that elevating the problem from targets-aircrafts level to chains-fleets level does not harm the constraints of the problem, yet it decreases the size of the problem.

The specific case study is analyzed and the necessary information for modeling of the problem is investigated. Moreover, procedure for carrying this information to the suitable structure to be used in the search methods is expressed. Since the problem is an assignment problem, the generic formulations of the assignment problems in the literature are investigated and these formulations are used for creating the base of our problem formulation. The problem of specific case study is formulated; the objective function and the constraints are discussed in details.

These kinds of problems are generally discussed in the literature as computationally costly problems where increase in the size of the problem exponentially creates handicaps in terms of solution time and memory requirements. Therefore, efforts have been paid for efficient implementation, algorithms, solution methods and heuristics.

Among many known methods, as the first solution method, Branch-and-Bound algorithm has been applied. Branch-and-Bound algorithm creates and searches a solution tree in which the branches and nodes having low or no possibility of having the solution are pruned. Branch-and-Bound algorithm is an exact solution method and searches the solution in hierarchical way with the use of the solution tree. A node on the solution tree holds a specific problem which might be the original problem or a sub problem of the original problem. The original problem is divided into sub problems by branching. This way, the solution space of the original problem is explored by dividing it into sub spaces. The algorithm branches to children nodes by making an assignment. The intelligence of Branch-and-Bound algorithm comes from its analysis on which nodes should be explored. Nodes and branches of the solution tree which do not have the potential to lead to the possible optimum solution are not included in the search and they are pruned. There are two pruning method. Nodes imposing infeasibility to the assignment are excluded from the search and they are pruned. The second pruning method of Branch-and-Bound algorithm is much more effective. At each node, a bound is calculated which is the estimate on the solution to the problem that node is representing. A bound is calculated by relaxing one or more constraints of the original problem. Moreover, the objective value of the most profitable and feasible assignment done so far is called as incumbent value. For our maximization problem, if the bound on a node is less than the incumbent value, then this node is pruned. Therefore, a more realistic bound and a better incumbent increase the performance of the algorithm. The disadvantage of Branch-and-Bound algorithm is the exponential growth of the solution tree with the problem size. For problems whose size is larger than 15 (i.e., number of chains in the scenario), this algorithm fails due to computational difficulties. Second known algorithm is Genetic algorithm in which elements of the solution space are mapped to the chromosomes and better solutions are searched by performing special operators imitating the evolution process of species. Genetic algorithm has no computational limits, but it may not return the global optimal assignment. However, after a certain number of generations, better solution can be found. These two algorithms are implemented for the specific case study and results are collected.

The results illustrated by the scenarios enable us to verify whether the implemented algorithms match their theoretical flows. We have observed that Branch-and-Bound algorithm fails for large-scale assignment problems. On the other hand, for small-scale problems, the global optimum assignment is obtained by Branch-and-Bound algorithm. We have implemented two version of Branch-and-Bound algorithm. In the first one, pruning of nodes is only based on feasibility criteria and this version is called as Branch-and-Bound – Global Search (BB-GS). Bounds on nodes are included in the second version in which bounds are used for optimistic estimation on the solution to the problem defined by the node. This version is called as Branch-and-Bound – Bounding Case (BB-BC), although it refers to the generic version of the algorithm. The initial incumbent in BB-BC is taken as zero and it is updates as the algorithm advances. With the use of bounds, it is expected to search the solution faster. We have observed that, bounds enable us to reach to the solution by evaluating fewer nodes. Yet, for large-scale problems, implemented versions of Branch-and-Bound algorithm fail to work as expected.

Genetic algorithm is best-known of its convenience for any type problems without causing computational complexities. Genetic algorithm has been implemented to the specific case study of fleet-chain assignment problem. Initial population has been created by random branching on the Branch-and-Bound search tree. Besides the known operators of Genetic algorithm, special features are developed. These special features carry infeasible solutions to the best feasible solutions around it. With the help of these special operations, an agile Genetic algorithm has been achieved which is eager to reach better solutions as soon as possible. The results of Genetic algorithm are investigated and it has been observed that the algorithm returns good assignments. Without any regard on its size, for assignment problems in which there are enough resources in the fleets to match all the chains of the scenario, implemented Genetic algorithm shows an utmost performance. The population has evolved rapidly at each generation and very good assignments are attained. On the other hand, for scenarios in which the number of resources in the fleets cannot suffice to match all the chains (the density of infeasibility is relatively high), the performance of GA slightly reduces. The increase in infeasibility causes difficulties to our special operators and the population cannot evolve as much as desired. Although the evolution does not occur at a desired level, Genetic algorithm has managed to find better solutions again.

At this point, it has been observed that Branch-and-Bound algorithm can return global optimum for small-scale problems, but fails for large-scale problems. Meanwhile, Genetic algorithm returns profitable assignments without regarding the size of the problem, yet better assignments could still be found. Therefore, we have decided to propose the manner for improving the results further. To do so, hybrid algorithms are defined which are using both Genetic and Branch-and-Bound algorithms.

The first hybrid algorithm is called as Hybrid I. The initial incumbent value has been assigned to zero in BB-BC. Since the initial incumbent value is not high, the performance is low considering the fact that a better incumbent will prune more nodes. Therefore, the profit of the fittest chromosome returned by Genetic algorithm is used as the initial incumbent to the BB-BC algorithm. As expected, the number of evaluated nodes decreases. The profit of the calculated assignment is higher than the one of BB-BC in most cases, so Hybrid I algorithm achieves to increase the performance. On the other hand, it does not increase the performance of the Branch-and-Bound algorithm that much; global optimum still cannot be found by Branch-and-Bound algorithm for large-scale problems.

The second proposed hybrid algorithm is called as Hybrid II. In its generic form, the root node of the Branch-and-Bound tree has no assignments within. The assignment of the fittest chromosome of Genetic algorithm is retrieved. This assignment is sorted according to the chain values. The single assignment of the most important chain is the first assignment of the assignment vector. A portion of this assignment vector is taken and this portion is declared as the assignment of the root node. Branch-and-Bound algorithm is then started from this node for searching the remaining unassigned chains. For scenarios in which all chains can be matched with fleets, this method is observed to fail, because all the single assignment whose alteration can improve the profit, are kept constant in the partial assignment of root node. On the other hand, for scenarios in which all chains cannot be assigned by a fleet, this method has been proven to work. For this particular type of scenarios (problems), Hybrid II algorithm obtains a more profitable assignment via the assignment of Genetic algorithm.

The third and the last hybrid algorithm is called as Hybrid III algorithm. The assignment of the fittest chromosome of Genetic algorithm is retrieved as the reference assignment. Then, Branch-and-Bound algorithm has started from a root node which has no initial assignment. A neighborhood (distance) notion is proposed which represents how dissimilar two assignments (branches) are. The search only evaluates the nodes and branches within a certain neighborhood of the reference assignment. The results of Hybrid III algorithm have been collected and it seems that Hybrid III is the best alternative for improving the results. For each tested scenarios, Hybrid III algorithm has managed to improve the results. The most profitable assignments returned by all implemented algorithms is the one calculated by Hybrid III algorithm, independently to the type of the scenario. Moreover, for distance threshold between 3 and 5, no computational difficulty has been faced. The performance of Hybrid III algorithm is superior compare to other algorithms and its performance does not depend on the conditions and constraints of the problem it is applied to.

7.2 Outcome of the Study

We believe that this thesis has been successful considering the objectives and the results of the study. In the beginning of the study, we have aimed to model the case study properly which we think we have achieved. We have tested the mostly used and known algorithms, Branch-and-Bound algorithm and Genetic algorithm, and we have observed that they are working as expected. Special feasibility operators of Genetic algorithm can be stated as a contribution. These special operators make the algorithm pretty agile, since each infeasible chromosome of the population is attempted to be directed to best feasible assignment around. The performance of Hybrid II algorithm has been observed to be dependent on the type of the problem. By this way, we have performed an analysis onto the relationship of the algorithms with the structure of the problem.

The best contribution of this study is Hybrid III algorithm. By this algorithm, better solutions are attained with little computational complexity. Hybrid III algorithm includes the best parts of Branch-and-Bound and Genetic algorithms while all disadvantages such as computational limitations are excluded. Without any regard on the structure of the problem, Hybrid III algorithm always find the most profitable assignments.

Since the computational complexity of the hybrid algorithms are totally dependent on their selected parameters like length of the partial assignment and distance threshold value, it is unfair to analyze their performance in terms of computational complexity. However, we can conclude that the computational complexity of Hybrid II algorithm converges to the complexity of the Branch-and-Bound algorithm as the selected length of the partial assignment decreases. Similarly, computational complexity of Hybrid III algorithm converges to the computational complexity of the Branch-and-Bound algorithm as the selected distance threshold increases. On the other hand, the analysis should not be on the computational complexity of the algorithms, we have rather focused more on the performance of the algorithms in a certain amount of time. Moreover, sharing of this certain amount of time between Genetic algorithm and Branch-and-Bound algorithm is another important topic.

7.3 Future Work

This study can be expanded in the following ways.

The case study can be expanded to the whole operation in which scheduling and simulation of the whole operation starting from preplanning and assignment problem to the target assessment and mission success assessment processes. For doing so, a stochastic structure should be designed for defining the mission failures of aircrafts. With the help of this stochastic structure, estimation of the future operation can be done. Since the full operation is going to be modeled, management of multiple sorties should be performed which is crucial for scheduling.

The study can be expanded to the scenario type specific assignment strategy in which the obtained assignment is highly dependent to the type of the scenario. Scenarios can be classified in terms of risk level and in risky scenarios more aircrafts are used than the chains require. Respectively, less number of aircrafts can be used for risk free scenarios. Moreover, in some scenarios there may be targets, assigning to which is crucial and a must. For these particular targets, ECM and RWR requirements which are for safe conduction of the operation can be relaxed in search algorithms. By this way, requirements of the operational order can be modeled and included in the algorithms.

The algorithms are implemented to the case study through a development environment where the choice of the algorithms and algorithm parameters are input hard-coded from this development environment. A graphical user interface can be developed for selection of algorithms and their parameters. Moreover, this GUI structure may also be used as interface for defining scenarios for testing.

One of the most convenient and applicable method for further improving the results and overcoming computational limitations is the use of parallel processing techniques.

REFERENCES

[1] E. M. Loiolaa, N.M.M. Abreua, P.O. Boaventura-Netto, P. Hahn and T. Querido, "An Analytical Survey for the Quadratic Assignment Problem," European Journal of Operation Research, Vol. 176, pp. 657-690, 2007.

[2] E. Çela, The Quadratic Assignment Problem: Theory and Algorithms, Kluwer Academic Publishers, 1998.

[3] L. Sagbansuai, "Algorithms for Very Large Scale Multi-Resource Generalized Assignment Problems," Business Administration: The University of Mississippi, 2004.

[4] A.M. Strouvalisa, I. Hecklb, F. Friedlerb and A.C.Kokossis, "An Accelerated Branch-and-Bound Algorithm for Assignment Problems of Utility Systems," Computers & Chemical Engineering, Vol. 26, pp. 617-630, 2002.

[5] X. Jiefeng, G. Bailey, "The Airport Gate Assignment Problem: Mathematical Model and a Tabu Search Algorithm," System Sciences, 2001.

[6] S. Rouillon, G. Desaulniers, F. Soumis, "An Extended Branch-and-Bound Method for Locomative Assignment," Transportation Research Part B: Methodological, Vol. 40, pp. 404-423, 2006.

[7] A.I. Giortzis, L.F. Turner, “A Mathematical Programming Approach to the Channel Assignment Problem in Radio Networks,” Vehicular Technology Conference, Vol. 2, pp. 736-740, 1996.

[8] T. Mizuike, Y. Ito, “Optimization of Frequency Assignment,” IEEE Transactions on Communications, Vol. 37, pp. 1031-1041, 1989.

[9] J. Garces-perez, D.A. Schoenefeld, R.L. Wainwright, “Solving Facility Layout Problems Using Genetic Programming,” MIT Press, pp. 28-31, 1996.

[10] S. Ramakrishnan, I.H. Cho, L.A. Dunning, “A Close Look At Task Assignment in Distributed Systems,” INFOCOM ’91, 1991.

[11] J.D. Clay, M.P. Wellman and C.R. Bisbee, “Temporal Reasoning for Airlift Scheduling Analysis,” NAECON ’90, Vol. 3, pp. 1181-1185, 1990.

[12] L.F. Bertucelli, M. Alighanbari, J.P. How, “Robust Planning for Coupled Cooperative UAV Missions,” IEEE Conference on Decision and Control, Vol. 3, pp. 2917-2922, 2004.

[13] T. Gürdal, “Harekat Alanında Muharip Jet Uçaklarının Hava Görev Emrinin Modellenmesi,” T.C. Kara Harp Okulu, 2005.

[14] N. W. Brixius, “Solving Large-Scale Quadratic Assignment Problems,” Department of Computer Science, The University of Iowa, 2000.

[15] P. Merz, B. Freisleben, “A Comparison of Memetic Algorithms, Tabu Search, and Ant Colonies for the Quadratic Assignment Problem,” Congress on Evolutionary Computation ‘99, Vol. 3, pp. -2070, 1999.

[16] L.M. Gambardella, E. Taillard, M. Dorigo, "Ant Colonies for the Quadratic Assignment Problem," The Journal of Operational Research Society, Vol. 50, pp. 167-176, 1999.

[17] D.L.H. Ramalhinho, D. Serra, "Adaptive Approach Heuristics for the Generalized Assignment Problem," Economics Working Paper 288, 1998.

[18] L. Chunguang, Y. Juebang, L. Xiaofeng, "Fuzzy Tabu Search for Solving the Assignment Problem," IEEE International Conferenceon Communications, Circuits and Systems, Vol. 2, pp.1151-1155, 2002.

[19] S. Haddadi and H. Ouzia, "Effective Algorithms and Heuristics for the Generalized Assignment Problem," European Journal of Operation Research, Vol. 153, pp. 184-190, 2004.

[20] P. Hahn, T. Grant, N. Hall, "A Branch-and-Bound Algorithm for the Quadratic Assignment Problem Based on the Hungarian Method," European Journal of Operation Research, Vol. 108, pp. 629-640, 1998.

[21] G. Kozanidis and E. Melachrinoudis, "A Branch-and-Bound Algorithm for the 0-1 Mixed Integer Knapsack Problem with Linear Multiple Choice Constraints," Computers & Operations Research, Vol. 31, pp. 695-711, 2004.

[22] C.L. Chen and S.C. Wang, "Branch-and-Bound Scheduling for Thermal Generating Units," IEEE Transaction on Energy Conversion, Vol. 8, 1993.

[23] P. Somol, P. Pudil and J. Kittler, "Fast Branch & Bound Algorithms for Optimal Feature Selection," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 26, 2004.

[24] S.J. Rasmussen and T.Shima, “Branch and Bound Tree Search for Assigning Cooperating UAVs to Multiple Tasks,” American Control Conference, 2006.

[25] “Equirectangular Projection”, http://wikipedia.org

[26] D. Delahaye, J.M. Alliot, M. Schoenauer, J.L. Farges, “Genetic Algorithms for Air Traffic Assignment,” Proceedings of European Conference on Artificial Intelligence, 1994.

[27] S. Chattopadhyay, P.P. Chaudhuri, “Genetic Algorithm Based Approach for Integrated State Assignment and Flipflop Selection in Finite State Machine Synthesis,” Proceedings of 11^{th} International Conference on VLSI Design, 1998.

[28] C.C. Chiu, C.H. Hsu, Y.S. Yeh, “A Genetic Algorithm for Reliability-Oriented Task Assignment with k~ Duplications in Distributed Systems,” IEEE Transactions on Reliability, Vol. 55, pp. 105-117, 2006.

[29] T.P. Fu, Y.S. Liu, J.H.Chen, “Improved Genetic and Ant Colony Optimization Algorithm for Regional Air Defense WTA Problem,” First International Conference on Innovative Computing, Information and Control, 2006.

[30] M. Alabau, L. Idoumghar, R. Schott, “New Hybrid Genetic Algorithms for the Frequency Assignment Problem,” IEEE Transactions on Broadcasting, Vol. 48, pp. 27-34, 2002.

[31] T.L. Lau, E.P.K. Tsang, “The Guided Genetic Algorithm and Its Application to the Generalized Assignment Problem,” Proceedings of 10^{th} International Conference on Tools with Artificial Intelligence, pp. 336-343, 1998.

[32] T. Shima, J. Rasmussen, A.G. Sparks, “UAV Cooperative Multiple Task Assignments Using Genetic Algorithms,” Proceedings of the American Control Conference, Vol. 5, pp. 2989-2994, 2005.

[33] D. R. S. Castro, “Optimization Models for Allocation of Air Strike Assets with Persistence,” Naval Postgraduate School, Thesis for Master Degree, 2002.

[34] D. G. Luenberger, “Linear and Nonlinear Programming,” 2nd Edition, Addison-Wesley Publishing Company, 1984.

[35] J. W. Chinneck, “Practical Optimization: A Gentle Introduction,” http://www.sce.carleton.ca/faculty/chinneck/po.html.

[36] “A* Search Algorithm”, http://wikipedia.org