Spring 2017

Optimal Ship Maintenance Scheduling Under Restricted Conditions and Constrained Resources

Vi Hong Nguyen
Old Dominion University, vnguy025@odu.edu

OPTIMAL SHIP MAINTENANCE SCHEDULING UNDER RESTRICTED CONDITIONS AND CONSTRAINED RESOURCES

by

Vi Hong Nguyen
B.S. June 2008, Hanoi University of Mining and Geology, Viet Nam

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirements for the Degree of

DOCTOR OF PHILOSOPHY

MECHANICAL ENGINEERING

OLD DOMINION UNIVERSITY
May 2017

Approved by:

Han Bao (Director)

Miltiadis Kotinis (Member)

Resit Unal (Member)

ABSTRACT

OPTIMAL SHIP MAINTENANCE SCHEDULING UNDER RESTRICTED CONDITIONS AND CONSTRAINED RESOURCES

Vi Hong Nguyen
Old Dominion University, 2017
Director: Dr. Han Bao

The research presented in this dissertation addresses the application of evolution algorithms, i.e. Genetic Algorithm (GA) and Differential Evolution algorithm (DE) to scheduling problems in the presence of restricted conditions and resource limitations. This research is motivated by the scheduling of engineering design tasks in shop scheduling problems and ship maintenance scheduling problems to minimize total completion time. The thesis consists of two major parts; the first corresponds to the first appended paper and deals with the computational complexity of mixed shop scheduling problems. A modified Genetic algorithm is proposed to solve the problem. Computational experiments, conducted to evaluate its performance against known optimal solutions for different sized problems, show its superiority in computation time and the high applicability in practical mixed shop scheduling problems. The second part considers the major theme in the second appended paper and is related to the ship maintenance scheduling problem and the extended research on the multi-mode resource-constrained ship scheduling problem. A heuristic Differential Evolution is developed and applied to solve these problems. A mathematical optimization model is also formulated for the multi-mode resource-constrained ship scheduling problem. Through the computed results, DE proves its effectiveness and efficiency in addressing both single and multi-objective ship maintenance scheduling problems.

Keywords: Mixed job shop scheduling, Mixed integer linear programming (MILP), Makespan, Preventive maintenance, Priority rules, Evolution Algorithm, Genetic Algorithm, Differential Evolution algorithm, Multi-objective optimization, Multi-mode resource-constrained scheduling problem.

This thesis is dedicated to the proposition that the harder you work, the luckier you get.

ACKNOWLEDGMENTS

First and foremost, I would like to express my heartfelt gratitude to my advisor, Professor Han Bao. As a tremendous mentor for me since the days I began working on my PhD program, Professor Bao has helped me in my transformation from being a timid student with limited English language skills to becoming a confident and competent student. With his insight, patience, motivation, and immense knowledge, he has nursed me from a student with poor knowledge in optimization into a research scientist. He has supported me not only by providing a research assistantship over almost five years but also he helped me overcome many difficulties during the research and writing of this thesis. I cannot imagine having a better advisor and mentor for my Ph.D. study.

Besides my advisor, I would like to thank the rest of my thesis committee, Professor Miltiadis Kotinis and Professor Resit Unal, for serving as my committee members, for their enthusiasm and their invaluable comments during the coursework that built my research background, and for their critique and encouragement to widen my thesis from various perspectives. I also wish to express my gratitude to Professor Duc Nguyen, Professor Mecit Cetin and Professor Jeremiah Creedon for allowing me to participate in their research groups, for their excellent teaching and for their encouraging way of leading me to deeper understanding and experience. Without their precious support, it would not be possible to conduct this thesis.

Many thanks to the Vietnam Education Foundation (VEF) funded by the United States government for inspiring, orienting and guiding me in my graduate admission to an American university. VEF created a launch pad with two financial support years for me to achieve my higher education dream. I am also grateful for the Department of Mechanical and Aerospace Engineering at Old Dominion University for providing me an excellent work environment during

my study. I thank the entire staff for their assistance and my friends who have helped me along the way in this long journey.

I cannot fully express my gratitude to my parents. Thank you for all the support, patience and for being there whenever I need you. I would not be where I am today without your constant love and encouragement.

Last but not least, a huge thank you to my beloved husband, Thanh Tran. You have just been incredibly supportive and committed to my success. Special thanks to my beloved daughters, Linh Tran and Vy Anh Tran for being such good girls and always cheering me up. I am so grateful to have them always by my side. I will never find words enough to express my deepest love and appreciation I have for them.

NOMENCLATURE

GA	Genetic Algorithm
DE	Differential Evolution
PDE	Pareto Differential Evolution
PMS	Preventive Maintenance System
MRO	Maintenance-Repair-Operation
CSP	Constraint Satisfaction Problem
CBR	Constraint -Based Reasoning
MOPs	Multi-objective Optimization Problems
MRCSSP	Multi-mode Resource-Constrained Ship Scheduling Problem
MSU	Maintenance Scheduling Utility
OBJ	Objective
ops	Operations
mtn	Maintenance activities
P	Total number of ships in the squadron/fleet
$st_c(p)$	Start time of the 1st maintenance activity in each cycle for ship
st(p,i)	The start time of the i^{th} maintenance activity
et(p,i)	The end time of the i^{th} maintenance activity
$d_{op}(p,i)$	The duration of the i^{th} operation activity ship p
$d_m(p,i)$	The duration of the i^{th} maintenance activity ship p
SAv(p,st_c (p),t)	Availability of ship p at any time t during the scheduling horizon
QAv(q,t)	Availability of squadron q

$FAv(t)$	Availability of a fleet consisting of Q squadrons
F	The mutation scaling factor
Cr	The crossover rate
$rand_p$	Randomly selected decision variable.
t	Type of maintenance activity
m	Mode
d_{tm}	Maintenance duration time type t mode m
$S1_{tm}, S2_{tm}, S3_{tm}$	Resources S1, S2, S3 type t mode m
$C1, C2, C3$	Cost for resources S1, S2, S3
$d_{ma}\ (p,i,m,t)$	Maintenance duration time of ship p, activity i, type t, mode m
$S1(p,w)$	Required resources S1 for ship p at week w
$S2(p,w)$	Required resources S2 for ship p at week w
$S3(p,w)$	Required resources S3 for ship P at week w
$Makespan(p)$	Required time to complete one cycle of ship p
$Makespan$	Required time to complete all ship actives of a squadron
T	The maximum planning makespan
M	A large number
U_a	Utility of ship availability,
U_m	Utility of makespan,
U_c	Utility of cost
w_j	The weight

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Ships play a critical role in the development of commerce, both national and international, and in national defense. They have developed alongside the human race and are major means of transportation over water. Historically, ships have been used for such purposes as colonization and slave trade, and have served scientific, cultural, and humanitarian needs. Nowadays, the development of the shipbuilding industry makes water transportation cheaper and more popular than ever. Ships can easily carry thousands of customers, capital goods, heavy machinery, and bulk raw and finished goods and so on from one place to another, between the same ports at time intervals based on a given schedule. They can also travel long distances or transport from and out of the country to foreign countries. According to the International Chamber of Shipping (ICS), over 90% of world trade is carried out by the international shipping industry. As of 2016, there were around 50,000 merchant ships trading internationally, transporting every kind of cargo [1]. The import and export of goods on the scale necessary for the modern world would not be possible without shipping. In addition, the continuous expansion of seaborne trade brings low and decreasing freight costs for consumers across the world, and shipping also is the safest and most environmentally benign form of commercial transport. Some types of ships are classified for commercial usage such as: tankers, cargo ships, carriers, ferries, and cruise ships [2]. Some are designed as naval vessels for combatant, auxiliary, coastal patrol and interdiction, carrier, and research. There are three categories of Modern naval vessels: surface warships, submarines, and support and auxiliary vessels. Modern surface warships

include aircraft carriers, frigates, cruisers, corvettes, destroyers, and amphibious assault ships. Submarines are watercrafts capable of independent operation underwater. They have a wide range of types and capabilities for different attacking and protection tasks. Support and auxiliary vessels also have many types, such as offshore patrol vessels, patrol boats, replenishment ships, minesweepers, and hospital ships, which are designated medical treatment facilities. Clearly, the quantity and quality of naval ships exert a tremendous influence over the power of combat and defense of a nation; therefore, nations with sea access consistently attempt to strengthen their naval fleet.

Besides the major role of ships as described above, some types of ships are designed for specialized usage such as: drill ships, which incorporate ship technology and sea plant technology; anchor handling and supply vessels for the offshore oil industry; salvage tugs; ice breakers; fishing ships; research vessels.

In the marine shipping industry, in order to maximize operating time during the ship's service life and to avoid breakdown and deterioration, effective maintenance scheduling is essential. In addition, due to its complexity and the obligations on shipping organizations to comply with certain regulations and requirements, ship maintenance scheduling has distinctive characteristics unlike the maintenance planning found in other industries. This chapter will review the need for maintenance, maintenance strategies, and key factors that affect maintenance scheduling. The ship maintenance scheduling will be discussed in detail to provide an understanding of its characteristics and the need for applying optimization in this type of scheduling.

1.1 Maintenance Scheduling Problem

Maintenance is a vitally important feature of the national economy. Each year, industrialized nations allocate large amounts of money to maintenance (Guignier and Madanat 1999) [3]. Poor or failed maintenance planning can result in serious incidents and high penalty costs arising from operation downtime (Dekker 1996) [4]. Maintenance not only keeps any mechanical equipment or machinery going but also can help with prolonged life and a favorable outcome. For a ship, maintenance is the one thing that keeps machinery in smooth running condition. White, E. N. (1979) [5] categorized maintenance by different strategies as indicated in Figure 1.

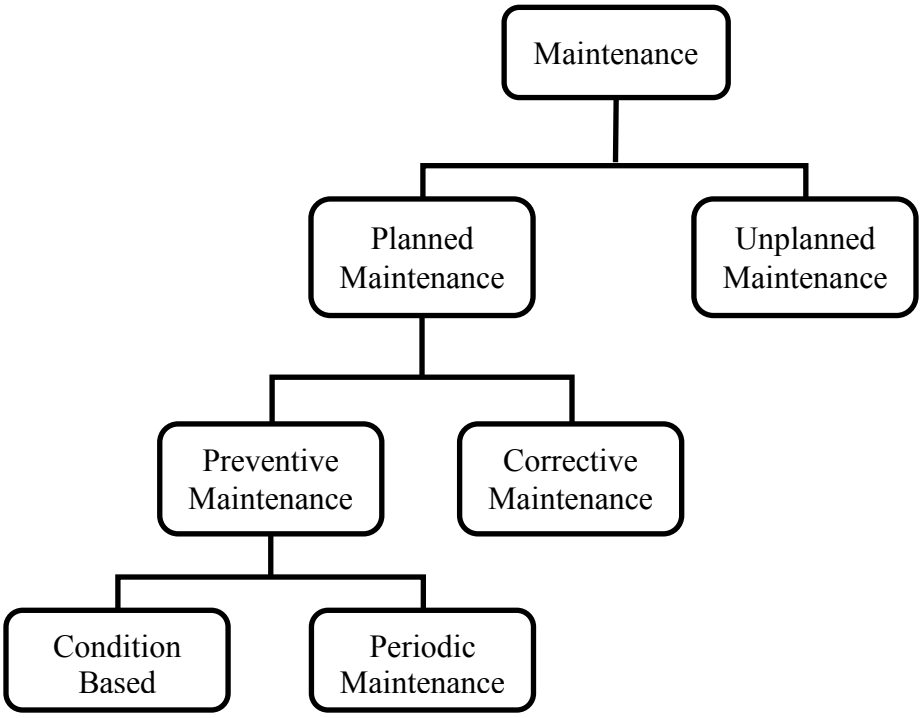

Figure 1. Maintenance Options (White 1979) [5]

There are two types of maintenance planning: scheduled (planned) maintenance and unscheduled (unplanned) maintenance.

1.1.1 Unscheduled (Unplanned) Maintenance

This type of maintenance is defined as breakdown maintenance or run to failure maintenance. Simply put, the repair or replacement is performed only when failure occurs.

Before World War II (1945) when industry was not highly mechanized and downtime had less serious effects, this approach was the standard approach [11]. However, analysis of maintenance costs indicates that repairs cost of this type average about three times higher than repairs made within a scheduled or preventive model because it results in high overtime labor, high machine downtime, low production, and high inventory costs for spare parts (Mobley 2002) [6]. Therefore, unscheduled maintenance is only suitable for a system with a low hazard rate and no serious cost or safety consequences.

1.1.2 Scheduled (Planned) Maintenance

Scheduled maintenance became more common after World War II (1945). In maintenance scheduling or planning, all necessary activities such as detecting failure and repairing or replacement are planned, controlled, and recorded in connection with keeping an installation to an acceptable standard (White 1979) [5] . Regular inspections and maintenance can prevent equipment failure, high costs and loss of time due to repair failing assets.

To protect the reliability and safety of a system, scheduled maintenance includes four basic task types (Nowlan and Heap 1978) [7]:

1. Inspection of a component to detect failure;
2. Failure detection;
3. Reworking and discarding of a component before its maximum age;
4. Inspecting an item to assess unseen failures.

For most systems, there are two classes of maintenance scheduling: preventive maintenance and corrective maintenance, one or both of which may be applied.

1.1.3 Corrective maintenance:

Corrective maintenance is carried out due to a breakdown and could be either planned or unplanned. Planned corrective maintenance is likely to be the result of a run-to-failure maintenance plan. The advantage of this system is that machinery parts are used to their full life or until they break. Unplanned corrective maintenance could be due to breakdown not stopped by preventative maintenance or a breakdown due to a lack of a maintenance plan. Unplanned maintenance, like reactive maintenance, is much more costly than planned maintenance because the unexpected breakdown may also damage other parts, especially in situations when a breakdown cannot be tolerated.

1.1.4 Preventive maintenance:

It is famously known as the PMS or Preventive Maintenance System. Scheduling repair or replacement time in preventive maintenance usually is built based on the manufacturer's recommendations and experience. In this case, the maintenance is performed within scheduled intervals like monthly, yearly, etc. or is carried out per the running hours like 4000 hours, 8000 hours etc. Maintenance is carried out irrespective of the condition of the machinery. The parts must be replaced if it is written in the schedule, even if they can be still used. Because parts are lubricants changed, adjustments, or replaced made before failure occurs, the objective is to increase the reliability of the system over the long term by staving off the age effects of wear, corrosion, fatigue, and relate phenomena. Preventive maintenance can be time-based or condition-based maintenance. Time-based preventive maintenance, as described above, is run

according to a certain time schedule. Condition-based preventive maintenance depends upon the monitoring of a given condition, for example the normal vibration arising from the operation of a piece of equipment. Whenever the vibration exceeds a certain threshold value then the equipment must be applied.

1.2 General ship maintenance practice

Ship maintenance is normally considered in the early stages of ship design (Shields et al. 1996) [8]. During the life cycle of a ship, configuration conversion, overhaul, and major repair work can be applied to maximize its operational readiness. While conversion/modernization change ships' configurations so that they can do better jobs or different jobs, the goal of overhaul is generally to bring them back to an almost like-new condition. The goal of repair is to maintain hull integrity of the ship and its equipment and to repair or replace damaged and worn parts. Figure 2 illustrates the basic actions carried out during maintenance that are significant during a ship's service life.

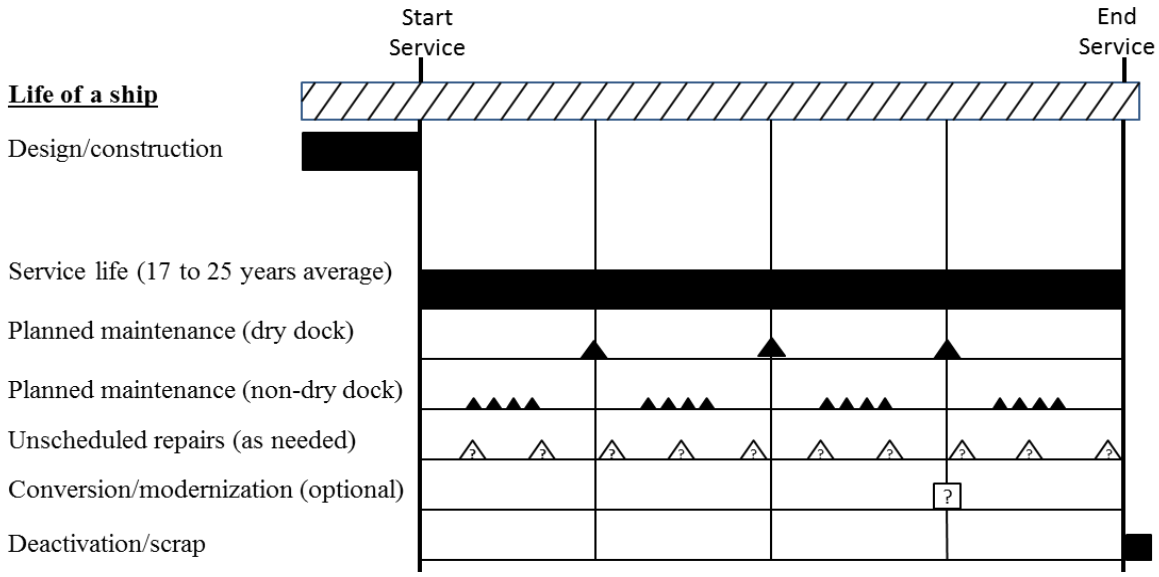

Figure 2. Maintenance/repair life of a ship (Richard Lee Storch 2007) [9]

Repair and maintenance of a ship generally falls into three categories discussed below.

1. Unscheduled voyage repairs. There can be many unscheduled repairs which become necessary whenever there is an unexpected equipment failure or fouling, a storm damage, a collision damage, a fire, or any other possibilities resulting in damage or breakdown occurring during operations. The size and complexity of unscheduled voyage repairs can range from very small and simple to very large and complex.

2. Planned maintenance is a regular maintenance program run by the owner based in his or her own maintenance philosophy, statistical information on equipment failure rates, or predictive data collection [9]. Planned maintenance can involve repair, replacement, or conversion/modernization for many different aspects of the ship, such as the ship's body or ship's surface, engine machinery, deck machinery, electrical equipment, navigation and communication systems, etc. All of these programs are designed to keep ships operating safely with a minimum of downtime, to significantly extend the life of the ship, and meet the latest safety standards.

3. Overhaul is a special larger-scale type of planned maintenance. Its purpose is to bring the overall operating condition of a ship back to good-as-new. Overhaul is a large and complex undertaking, lasting from a few months to a year for many military ships [9].

At the end of the economical service life of a ship, the owner has to determine how to dispose of it (deactivation/scrapping). Some ships, especially in the military, are deactivated and stored for possible later use. Others are broken up for the scrap or resale value of their materials and equipment [9].

Ship maintenance can be conducted in different locations as is presented in Figure 3.

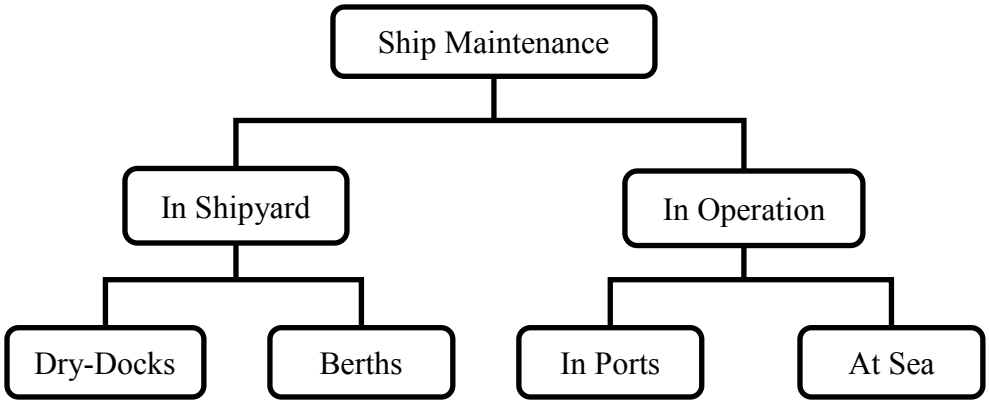

Figure 3. Locality of Ship Maintenance (Yousef Alhouli 2011) [11]

The maintenance can be conducted in the shipyard, where the ship must be dry docked, when major overhaul is needed. The use of different shipyards may be required by different types and sizes of ships (Deris et al. 1999) [12]. In the shipyard, typically 75% of the work involves routine ship maintenance, and the remaining 25% is for damage repair and ship conversion (Mackenzie 2004) [13]. It also can be conducted at anchorage and in the harbor (non-dry dock) when medium maintenance is needed. Minor maintenance can be done during a ship's daily operations. Non-dry dock can be pier-side inspection and repair involving topside work where a dry dock is not needed for access. This kind of maintenance is done frequently (usually annually). Dry-dock maintenance is a major maintenance involving inspection and repair of the underwater hull, propellers and shafts, rubber, thrusters, hull coatings, cathodic protection, sea chests, and other underwater items; usually it also includes all the work done in a non-dry dock maintenance and small to medium sized configuration changes planned for the ship (including overhauls) [9]. Non-dry dock maintenance is done periodically.

1.3 Motivation of dissertation

Minimizing the out-of-service time for a ship is extremely important for its owner because out-of-service time is very expensive. Rapid completion of repair work may be more important than a low repair cost for a business. In the highly competitive business of ship repair and modernization, a good maintenance plan can bring many advantages.

Firstly, it increases the availability of ships. Every ship is provided with a maintenance cycle and every component in the ship is scheduled to be maintained within the maintenance scheduling plan to maximize the ship's availability. The ship is considered available if all its major systems, such as propulsion, power, air-conditioning, and cargo machines, is in full operational readiness. The ship will be classified as unavailable, if any one of the major components is defective, and maintenance will be required. Maintaining high availability can only be achieved by effective maintenance management of which maintenance scheduling is considered as one of the main factors (Deris et al. 1999) [12].

Secondly, it optimizes use of resources. Resource allocation is a key for efficient management and operation. Planning of maintenance helps with the use of available resources such as labor, equipment, and shipyards, avoid overload, re-work, or waste resources. Moreover, maintenance teams have less unplanned maintenance and can respond quicker to new problems. This decreases MRO (Maintenance-Repair-Operation) cost, and increases profit.

Thirdly, it facilitates meeting deadlines and requirements. Ship maintenance is scheduled in the order of priority of time and importance for each ship. This avoids delay in deploying the project or meeting the requirement of customers.

Fourthly and finally, it leads to maintaining the proper quality of ships. For regular maintenance, each kind of ship has its own maintenance cycle. To ensure extending the useful

lifecycle of ship and decreasing the need for capital replacements, it is maintained through its maintenance cycle. During the regular maintenance, small problems can often be detected before they become large problems with expensive repairs; technicians can anticipate failures or identify problems early that have potentially hazardous conditions in the future. This not only enhances efficiency and reliability of ships but also improves the performance of ship: increasing uptime, reducing downtime but also reducing repair cost and avoiding overhaul operations.

Understanding the extremely important role of ship maintenance scheduling so that operations remain safe over a long-time cycle, the application of any optimization scheme to this problem is well worth the effort and has been a fertile field for researchers in recent years.

In the remainder of this dissertation, chapter 2 provides a brief review of the literature of maintenance scheduling and the research foundation for this dissertation. Chapter 3 deals with the problem of mixed shop scheduling. Chapter 4 provides the foundation for applying the Differential Evolution (DE) approach to the Navy Ship Maintenance Scheduling Problem including details on the programming aspects and solution. Chapter 5 presents an overall conclusion for this dissertation and suggestions for future work.

CHAPTER 2

LITERATURE REVIEW AND RESEARCH FOUNDATION

This chapter gives a brief literature review of the optimization of maintenance scheduling. It also gives a summary of fundamentals of genetic algorithm (GA) and differential evolution (DE), and their applications based algorithms and limitations are outlined. Some basic concepts of GA and DE are presented. After that, general block diagrams of GA and DE are provided to indicate how the optimization problems can be solved.

2.1 Maintenance scheduling optimization literature review

Maintenance scheduling optimization has been studied extensively from the early 1960s with the development of many maintenance optimization models. Among these models, the integer linear programming technique was found to be the most commonly used model.

Mukerji et al. (1991) [14] discussed a number of different optimization goals and optimization techniques in their maintenance schedules for the Wisconsin Power and Light Company. They used a mixed-integer program to solve the maintenance scheduling problem that attempted to level the generating reserves throughout the planning horizon.

A mixed-integer linear programming model was developed by Ashayeri et al. (1996) [15] to simultaneously plan preventive maintenance and production in a process industry environment, where maintenance planning is extremely important. The paper schedules production jobs and preventive maintenance jobs, while minimizing costs associated with production, backorders, corrective maintenance and preventive maintenance

Deris et al. (1999) [12] modeled ship maintenance scheduling as a constraint satisfaction problem (CSP). The variables used in the model were based on the start times, and the domain

values were the start and the horizon of the schedule. These authors applied their model to the Royal Malaysian Navy. They adopted a constraint-based reasoning (CBR) together with a genetic algorithm (GA) to find the start times of the first maintenance activity of each ship to maximize its availability or the availability of a squadron of ships, or an entire fleet for operations that satisfied maintenance requirements, dockyard availability, and operational requirements.

Baliwangi et al. (2006) [16] presented a ship maintenance scheduling management system integrated with a risk evaluation and lifecycle cost (LCC) assessment approach. The approach improves upon existing practices in arranging an optimal maintenance schedule by modeling operational and economical risks. It analyzed risks associated with some operational problems such as operating schedule, routes, ship position, resource availability, and achievement of reliability-availability-maintainability (RAM) of system, determining component function, generating the time predicted and possible component combinations, and selecting the best alternative based on LCC.

Charles-Owaba et al. (2008) [17] established a new approach for evaluating the sensitivity of a preventive maintenance scheduling model which is based on an integrated operations maintenance activity schedule in a resource-constrained environment, and they tested it on a shipping company. Their results show that some shipping maintenance scheduling parameters are sensitive and therefore could be manipulated for the best performance of maintenance scheduling models

This thesis adopts the premise of the ship maintenance scheduling problem from the paper “Theory and Methodology of Ship maintenance scheduling by genetic algorithm and constraint-based reasoning” by Safaai Deris, Sigeru Omatu, Hiroshi Ohta, Lt. Cdr Shaharudin

Kutar, Pathiah Abd Samat [12]. In the next chapter, a modified genetic algorithm (GA) is developed to find the optimal start times of ship maintenance activities that maximize the availability of different squadron of ships and the utility of given shipyards. The problem is further expanded to multi objective optimization with a limited number of dockyards and is solved by a differential Evolution (DE) algorithm. A continuation of the DE solutions for the more practical problems involving resources constraints and mode assignments is also a part of this dissertation.

2.2 Genetic algorithm

2.2.1 Genetic algorithm literature review

Genetic algorithms were first proposed by Fraser [18], [19] and later expanded by Bremermann and Reed *et* al [20] and are considered possibly be the first algorithmic models developed to simulate genetic systems. Some basic concepts have been proposed and applied to engineering systems in their work. It was the extensive work done by Holland [21] that concepts of chromosomes, genes and fitness functions have now become standard features in GA. Because of his work, Holland is nowadays considered as the father of GA. In recent years, GA's population-based approach [22], [23] have a significant contribution in optimization problems in which the ability to make pair-wise comparison in tournament selection is exploited to devise a penalty function approach that does not require any penalty parameter. Feasible and infeasible solution comparisons are made so as to provide a search direction towards the feasible region. Once sufficient feasible solutions are found, a niching method [23] (along with a controlled mutation operator) is used to maintain diversity among feasible solutions. This allows a real-parameter GA's crossover operator to continuously find better feasible solutions which gradually lead the search near the true optimum solution. GAs with this constraint handling approach [23]

have been tested on many problems commonly used in the literature, including engineering design problems. With the rapid development of digital computers in recent years, the genetic algorithm has continued to have a major impact on optimization problems and intelligent system design problems. For example, Aytekin Bagis (2007) [24] used it as the optimization algorithm to combine a classical genetic algorithm structure with a systematic neighborhood structure.

2.2.2 Applications and Limitations of Genetic Algorithms

Genetic algorithms can be applied to a wide variety of optimization problems such as modern automation engineering, scheduling, computer games, stock market trading, transportation networks, vehicle routing, and medical applications.

Although the generic algorithm is able to find a good solution to a problem where the iterative solution is too prohibitive in time and the mathematical solution is not attainable in a fast manner, many limitations are encountered in the use of this approach such as robustness and convergence. For example, stop criterion is not clear in every problem, GA is sensitive to the initial population used or GAs may have a tendency to converge towards local optima or even arbitrary points rather than the global optimum of the problem. However, with the rapid development of digital computer, the applications of the GA get much attention and have achieved many important outcomes in solving the engineering problem.

2.2.3 General genetic algorithm

Being motivated by the principles of natural genetics and natural selection, GA is a stochastic global search method that works on populations of individuals instead of single solutions. It starts with the initial population that has no knowledge of the correct solution. Then GA searches in parallel and depends completely on responses from its evolution operators, i.e.

crossover, mutation, and reproduction, to arrive at the best solution. Figure 4 shows the basic structure of a genetic algorithm.

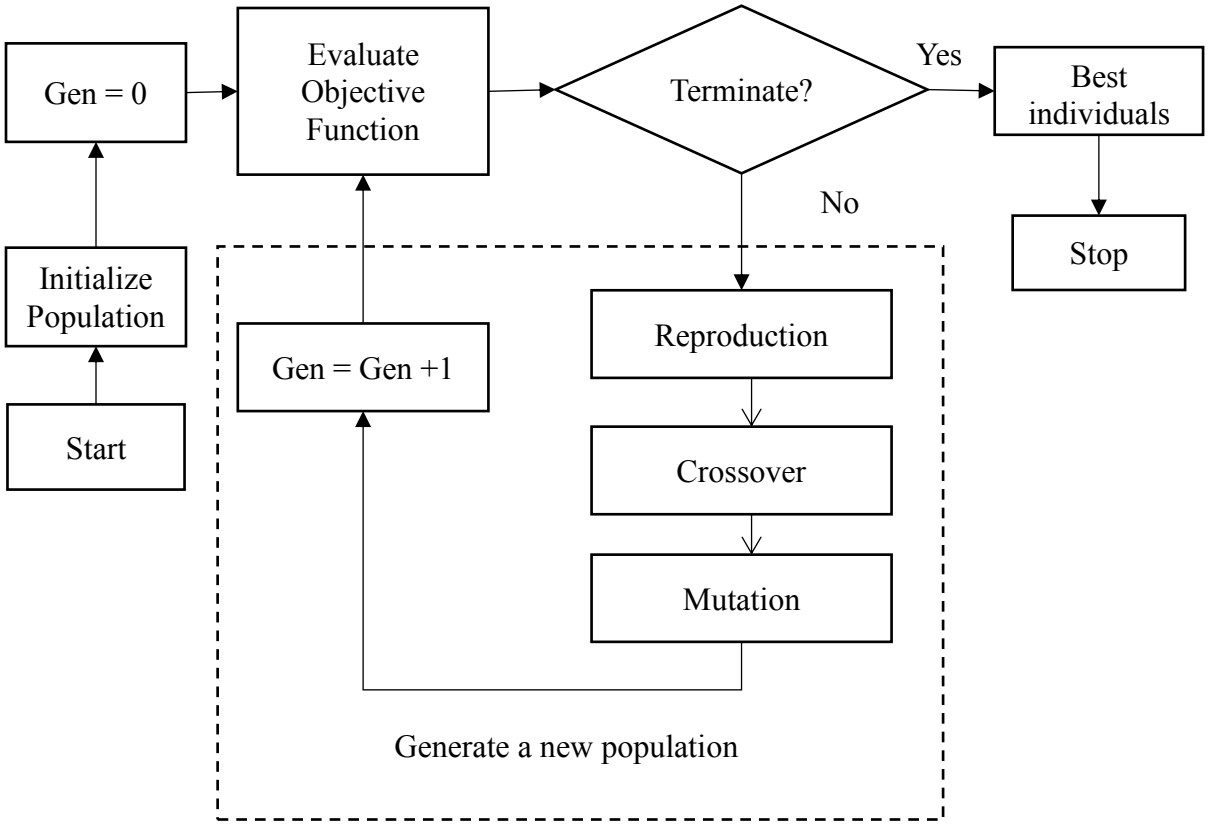

Figure 4. Block diagram of GA

2.2.3.1 Initial Population

The candidate solution to a problem is called a chromosome. The first step on applying a GA optimization is to generate an initial population (a collection of chromosomes). The standard way of generating an initial population is random selection. Initial population should be a uniform representation of the entire search space. Otherwise, there are regions of the search space that are not covered by the initial population. Consequently, they can be neglected by the search process. The size of the initial population depends on the computational complexity and

exploration abilities. Then GA evolves the population through multiple generations by using the genetic operators, i.e. reproduction, crossover, and mutation, in the search for a good solution. Some definitions are provided to introduce the block diagram of the GA for engineering system applications.

2.2.3.2 Objective function/Fitness Function

A mathematical function is used to quantify how good the solution represented by a chromosome is. This is to determine the ability of an individual of a GA to survive because individuals with the best characteristics have the best chance to survive and to reproduce according to the Darwinian model of evolution. The fitness function measures the quality of the represented solution by mapping a chromosome representation into a scalar value or represents the objective function that describes the optimization problem.

2.2.3.3 Selection

Selection is one of the main operators in GA that emphasizes the creation of a better generation and reproduction of chromosomes. There are two main steps of a GA selection:

- Selection of the new population: This occurs at the end of each generation. There a new population of candidate solution is selected from both the parents and the offspring, or from only the offspring to serve as the population of the next generation.
- Selection of parents for reproduction: The process of producing offspring from selected parents by applying crossover and/or mutation operators is also very important. To ensure that offspring contain genetic materials of the best individuals, better individuals should have more opportunities to reproduce. However, weak individuals can also increase their chances of survival by mutation, which may result in introducing better traits to weak individuals. Many selection operators have been developed for GA such as

random selection, proportion selection, tournament selection, rank - based selection, Elitism, Hall of Fame, Boltzmann Selection, etc. This part will discuss three of the most popular selection operators.

Random Selection: No fitness information is used in random selection. Each individual has the same probability to be selected which means that the best and the worse individuals have the same probability of surviving to the next generation.

Proportional Selection: Biased selection towards the most fit individuals. A probability distribution proportional to the fitness is created, and individuals are selected by sampling the distribution. The popular sampling methods used in proportional selection is roulette wheel sampling [25]. In roulette-wheel sampling, the implementation of the proportionate selection is thought of a roulette-wheel mechanism. The wheel is divided into N (population size) divisions, where the size of each is marked in proportion to the fitness of each population member [26]. The wheel is spun N times, each time choosing the solution indicated by a pointer. In this way, the individual with higher fitness value has a higher probability of being copied into the mating pool or surviving to the next generation.

Tournament Selection: In the tournament selection, a group of n individuals randomly is selected from the population, where n< N (N is the total number of individuals in the population). The fitness values of the selected n individuals are compared, and the best individual from this group is chosen and placed in the mating pool for reproduction or to the next generation.

2.2.3.4 Crossover

After selection is finished, the crossover procedure is initiated. Crossover is a genetic operator that produces a new chromosome (offspring) by combining (mating) two chromosomes

(parents). Crossover occurs during evolution according to a user-definable crossover probability. The idea behind crossover is that if the new chromosome takes the best characteristics from each of the parents, it may be better than both parents. However, the selection pressure may cause premature convergence due to reduced diversity of the new populations if selection focuses on the most-fit individuals [12]. There exists a number of crossover operators in the GA literature. They can be divided into three main categories [25]:

- **Asexual**, where an offspring is generated from one parent.
- **Sexual**, where two parents are used to produce one or two offspring.
- **Multi-recombination**, where more than two parents are used to produce one or more offspring.

Three basic crossover operators are presented below.

- **One-point crossover:** A one-point crossover operator was developed that randomly selects a crossover point and the segments after that point are swapped between the two parents. One-point crossover is illustrated in Figure 5.

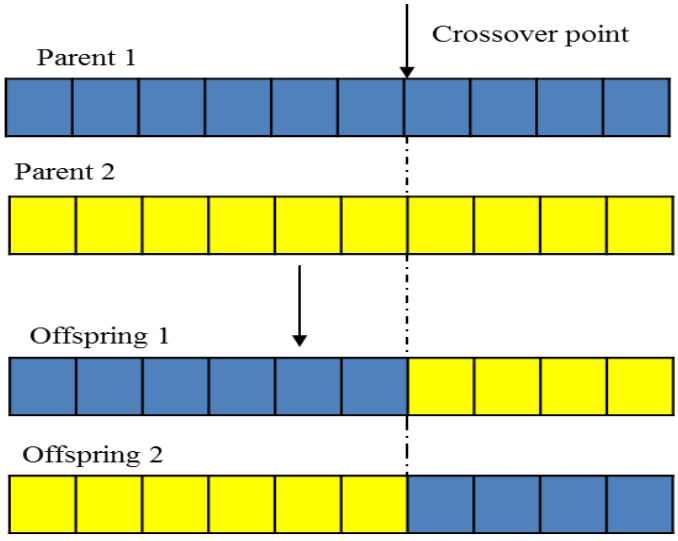

Figure 5. One-point Crossover

• **Two-point crossover:** A crossover operator that randomly selects two crossover points within a chromosome then interchanges the two parent chromosomes between these points to produce two new offspring. In this case two bit positions are randomly selected, and the segments between these points are swapped as illustrated in Figure 6.

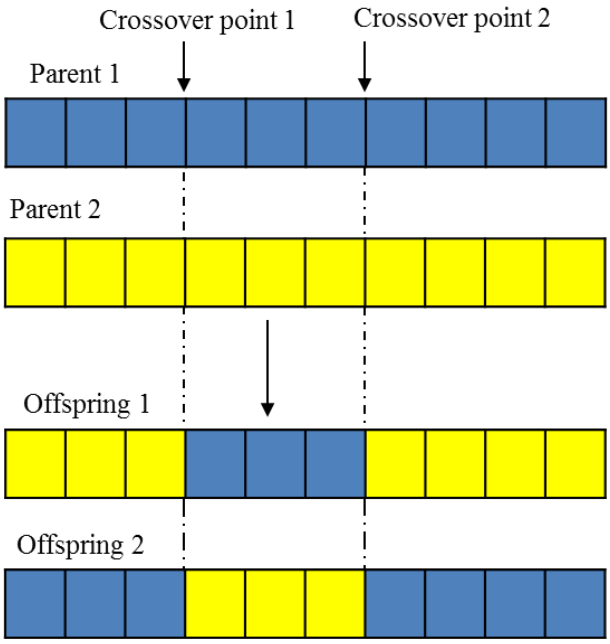

Figure 6. Two-point Crossover

• **Uniform crossover:** Uniform Crossover uses a fixed mixing ratio between two parents. Unlike one- and two-point crossover, the Uniform Crossover enables the parent chromosomes to contribute the gene level rather than the segment level. Uniform crossover is illustrated in Figure 7.

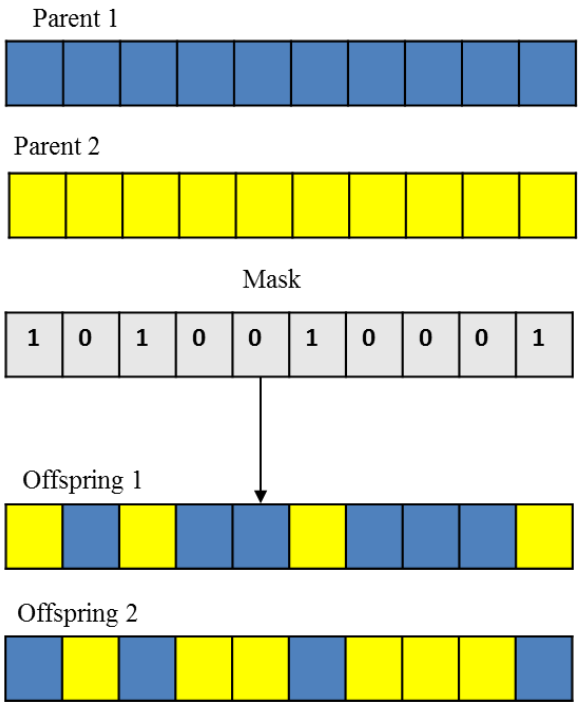

Figure 7. Uniform Crossover

2.2.3.5 Mutation

Mutation is an important operator of the genetic search as it helps to prevent the population from premature convergence at any local optima. Selection and crossover can create better solutions in the population. However, there may not be enough diversity in the initial population to ensure that the GA searches the entire problem space if GA only uses reproduction and crossover operators. Moreover, a bad choice of initial population can also make GA converge on sub-optimum. In mutation, the values of genes in a chromosome are randomly changed. This can result in the introduction of new genetic materials into the population, therefore increasing genetic diversity. With these new gene values, individuals may become better than the previous ones. Mutation probability, also referred to as the mutation rate, is defined by a user and should usually be set fairly low to ensure that good solutions are not

distorted too much. The high mutation probability can lead the search turn into a primitive random search.

2.2.3.6 Termination

The GA operators are iteratively applied until the stopping criterion is satisfied. There are many criterions for terminations. The simplest is to limit the number of generations that GA is allowed to execute. This limit should be big enough; otherwise, the GA will not have sufficient time to explore the entire search space. Another criterion that can be used is population convergence. If there is no change or small change in genotypic information of the population over a number of consecutive generations, GA can be stopped.

From all above concepts, the GA for programming is as follows:

Genetic Algorithm:

Initiate the strategy parameters

Create and initialize the initial gains

For each gain

Evaluate the objective function J

End

While stopping condition(s) not true do

For i =1, n do

Choose $i \geq 2$, new gains at random

Create offspring through application of crossover operator

Mutate offspring strategy parameters

Evaluate the objective function of new gains

If value of objective function is less than epsilon

Best gains

End

End

Select the new population

t = t +1

End

2.3 Differential Evolution

2.3.1 Differential Evolution literature review

Differential evolution (DE) is another kind of evolutionary algorithm developed by Storn and Price in 1995. At the First International Contest on Evolutionary Optimization in May 1996, the success of DE was demonstrated. Unlike GA, DE does not require the transformation of variables into binary strings, and it usually takes less computational time than GA. DE requires fewer parameter settings and is able to solve high-dimensional complex optimization problems. Since 1996, many researchers have tried to improve the performance of the basic differential evolution algorithm in several scientific problems and applications. The DE inventors, Storn and Price, wrote several papers about the basis of DE, highlighting the advantages of DE in solving difficult optimization problems including global optimization problems which subject to multiple nonlinear constraints [[30], [31], [32], [33], [34], [35]]. Lampinen (2002) proposed a constraint handling approach for the differential evolution algorithm. An extension for the Differential Evolution algorithm for handling nonlinear constraint functions was performed. He modified the replacement criterion of the original algorithm by applying a new replacement criterion for handling the constraints. The approach was demonstrated by solving ten well-known test problems [36]. Gamperle et al. (2002) studied different parameter settings for DE on several uni-

modal and multi-modal test functions. The article guided and provided the appropriate control parameters: the mutation scale factor F, the crossover constant Cr, and the population size Np. They concluded that the choice of control parameters Np, F, and Cr have a strong effect on the capability of finding the global optimized point and the convergence of DE [37]. Liu and Lampinen (2005) introduced a new version of Differential Evolution algorithm with adaptive control parameters – the fuzzy adaptive differential evolution algorithm. In fuzzy adaptive differential evolution algorithm, fuzzy logic controllers are used to adapt the search parameters for the mutation operation and crossover [38]. Mezura-Montes et al. (2006) presented a Differential-Evolution based approach to solve constrained optimization problems. The approach allowed each solution to generate more than one offspring but using a different mutation operator which combines information of the current parent and information of the best solution in the population to find new search directions. This increased the probability of each parent to generate a better offspring. Three selection criteria based on feasibility are used to deal with the constraints of the problem, and a diversity mechanism is added to maintain infeasible solutions located in promising areas of the search space [39]. Mallipeddi and Suganthan (2008) investigate the effect of population size on the quality of solutions and the computational effort required by the DE Algorithm. They used a set of five benchmark problems to study the effect of population sizes on the performance of the DE. The study showed a significant influence of the population size on the performance of DE, interactions between mutation strategies, population size as well as dimensionality of the problems [40]. Qin et al. (2009) propose a self-adaptive DE (SaDE) algorithm. In the proposed algorithm both trial vector generation strategies and their associated control parameter values are gradually self-adapted by learning from their previous experiences in generating promising solutions [41]. An alternative differential evolution algorithm for solving

unconstrained global optimization problems was presented by Mohamed et al. (2012). They introduced a new directed mutation rule based on the weighted difference vector between the best and the worst individuals of a particular generation to enhance the local search ability of the basic of the basic DE and to increase the convergence rate, and two new scaling factors as uniform random variables to improve the diversity of the population and to bias the search direction. A dynamic non-linear increased crossover probability scheme is utilized to balance the global exploration and local exploitation. They merge a random mutation scheme and a modified Breeder Genetic Algorithm mutation scheme to avoid stagnation and/or premature convergence [42].

2.3.2 Applications and Limitations of Differential Evolution

Differential evolution has been successfully applied in many applications on various domains such as neural network learning, digital signal processing, image processing, aerodynamic shape optimization, automated mirror design, and mechanical engineering design, physics, computer science, shape, control science, traffic control, manufacturing, management and even economics.

Although by running several vectors simultaneously, DE runs more stable than GA, DE is still easy to drop into a regional optimum. As GA, DE also cannot guarantee an optimal solution. The performance and results of the algorithm depend on the parameter settings (the mutation scale factor, the crossover constant, and the population size), and their configuration is sometimes an art rather than a science.

2.3.3 Basic Differential Evolution

Although DE shares similarities with GA, it differs significantly in the sense that distance and direction information from the current population is used to guide the search process. For GA covered in the previous part, variation from one generation to the next is achieved by applying crossover and/or mutation operators. If both of these operators are used, crossover is applied first, after which the generated offspring is mutated. DE differs from GA in that mutation is applied first, then crossover operator is used to produce an offspring; while for GA, mutation step sizes are sampled from some probability distribution function, mutation step sizes in DE are influenced by differences between individuals of the current population (Andries P.Engelbrecht 2007).

The basic components of the DE algorithm are explained below using Figure 8 as a block diagram to link these components together.

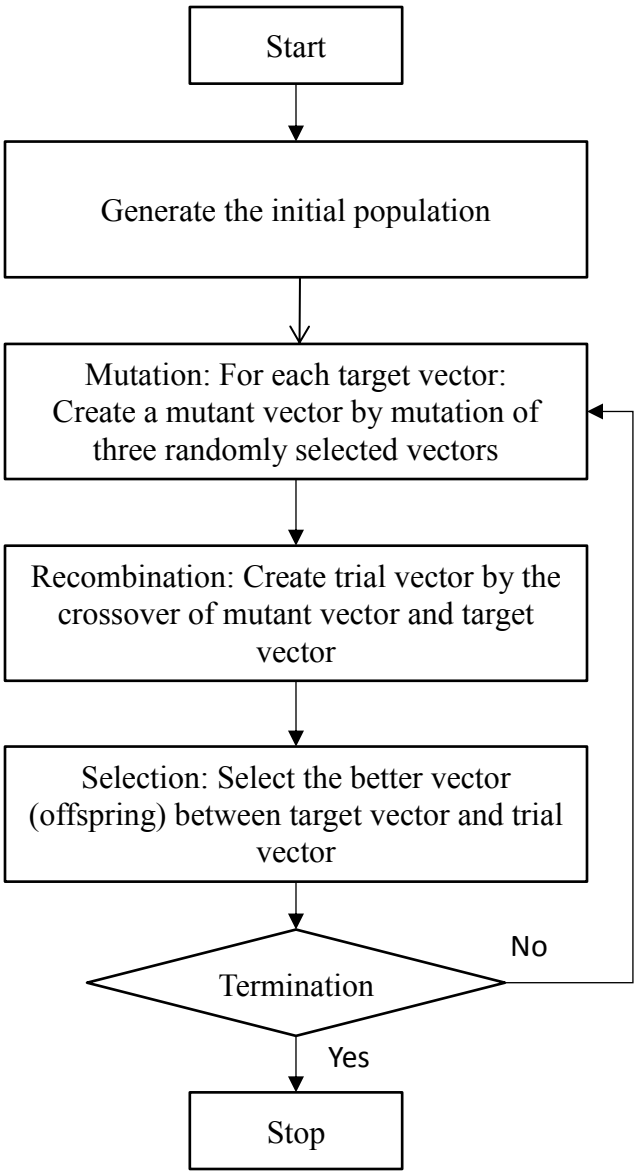

Figure 8. Block diagram of the DE

Initialization

DE is a population-based optimizer that creates the initial population by multiple, randomly chosen initial points. The population, symbolized by $P_{\mathbf{x}}$, is composed of the N_p vectors, $\mathbf{x}_{i,g}$. The index, $g = 0, 1, ..., g_{\max}$, indicates the generation to which a vector belongs. In addition, each vector is assigned a population index, i, which runs from 0 to $N_p - 1$, where N_p is

the number of vectors in the population. Parameter bounds must be specified before the population can be initialized to define the domain from which the N_p vectors in the initial population are chosen.

Mutation

Once initialized, DE mutates and recombines the population to produce a population of *Np* trial vectors. The DE create a mutant vector $v_{i,g}$ by combining three different, randomly chosen vectors as follows: for each parent, $\mathbf{x}_{i,g}$, select a target vector x_{r0} from the population such that $i \neq r0$. Then, randomly select two individuals, x_{r1} and x_{r2} from the population such that the indices $i, r0, r1, r2$ are distinct ($i \neq r0 \neq r1 \neq r2$)

Then adding the weighted difference of two of the vectors to the third as the equation below:

$$v_{i,g} = x_{r0,g} + F.\left(x_{r1,g} - x_{r2,g}\right)$$

The scale factor, $F \in (0,1)$, is a positive real number that controls the rate at which the population evolves. While there is no upper limit on *F*, effective values are seldom greater than 1.0 [32].

Crossover

The DE crossover operator implements a uniform crossover, sometimes referred to as discrete recombination of the mutant vector; $v_{i,g}$ and the parent vector; $\mathbf{x}_{i,g,}$ to to build a trial vector $u_{i,g}$ as follows:

$$u_{i,g} = u_{j,i,g} = \begin{cases} v_{j,i,g} \; if \; \left(rand_j(0,1) \leq Cr \; or \; j = j_{rand}\right) \\ x_{j,i,g} \; otherwise. \end{cases}$$

where $x_{j,i,g}$ refers to the *j*-th element of the vector $\mathbf{x}_{i,g}$, Cr is the crossover probability. It is a user-defined value, $Cr \in [0,1]$, that controls the fraction of parameter values that are copied from the mutant. To determine which source contributes a given parameter, uniform crossover compares *Cr* to the output of a uniform random number generator, $rand_j$ (0,1). If the random number is less than or equal to *Cr*, the trial parameter is inherited from the mutant, $v_{i,g}$; otherwise, the parameter is copied from the vector, $\mathbf{x}_{i,g}$. In addition, the trial parameter with randomly chosen index, j_{rand}, is taken from the mutant to ensure that the trial vector does not duplicate $\mathbf{x}_{i,g}$.

Selection

To determine which parent or offspring will survive to the next generation, the target (parent) vector $x_{i,g}$ is compared with the trial (offspring) vector $u_{i,g}$ .If the trial vector has an equal or lower objective function value than that of its target vector, it replaces the target vector in the next generation; otherwise, the target is admitted to the next generation.

$$x_{i,g+1} = \begin{cases} u_{i,g} \; if \;\; f(u_{i,g}) \leq f(x_{i,g}) \\ x_{i,g} \; otherwise. \end{cases}$$

General DE algorithm

Set the generation counter, t=0

Initialize the control parameters, F and Cr;

Create and initialize the population, *G(g)*, of N_p individuals;

While stopping condition(s) not true do

for *each individual*, $x_{i,g} \in G$ do

Evaluate the fitness, $f(x_{i,g})$;

Create the mutant vector, $v_{i,g}$ by applying the mutation operator;

Create a trial (offspring) vector $u_{i,g}$ by applying the crossover operator;

if $f(u_{i,g})$ *is better than* $f(x_{i,g})$ then

Add $u_{i,g}$ to G (g+ 1);

end

else

Add $x_{i,g}$ to G(g+ 1);

end

end

end

The solution is the individual with the best fitness.

2.4 Research approach of dissertation

The aim of this thesis is to develop and apply a combination of GA and DE procedures directly to scheduling problems in order to efficiently solve them. The strength of GA and DE lies in their ability to find a good solution to a problem where the mathematical solution is not attainable. They can find the solution in a fast manner and very effectively. A number of recent studies comparing DE with GA indicate that DE displays better and more stable results than GA [[43], [44], [45], [46], [47]]. In contrast to GA, where parent solutions are selected based on fitness, every solution in DE takes turns being a target vector (one of the parents). Therefore, all vectors play a role as one of the parents with certainty. The second parent is the mutant vector, which is formed from at least three different vectors. Thus, the trial vector is formed from at least

four different vectors and would replace the target vector only if this new vector is better than the target vector; otherwise, it would be ejected. This replacement takes place immediately without having to wait for the whole population to complete the iteration. Then, this improved vector would immediately be available for random selection of vectors to form the next mutant vector. Moreover, in DE, a new solution would be selected only if it has better a fitness. Therefore, the average fitness of the population would be equal or better than the average fitness of the previous population from iteration to iteration. Any improvement in the solution is immediately available to be randomly selected for the next target vector. This is different from GA where an improvement would take effect only after all the solutions have completed the iteration.

While DE is proved to be much more robust in many experiments, many researchers also mentioned that GA can produce as good results for small sized problem as DE and the running time of GA is comparable and in many cases better than DE [[43], [44], [45], [46], [47]]. Therefore, both GA and DE will be studied in this thesis through two case studies: a mixed shop scheduling problem and a maintenance scheduling problem specific to Navy ships and their particular requirements. The mixed shop scheduling is introduced, and a modified GA is proposed to solve it in chapter 3. When compared with previous methods, the proposed GA has achieved solutions with good accuracy, stable convergence characteristics, and lesser computation time. Chapter 3 also presents the MATLAB code using the modified GA procedures and a detailed comparison. In chapter 4, the Navy Ship Maintenance Scheduling Problem is presented and a DE approach is applied to solve it.

CHAPTER 3

AN EFFICIENT SOLUTION TO THE MIXED SHOP SCHEDULING PROBLEM USING A MODIFIED GENETIC ALGORITHM

The shop scheduling problem (SSP) is one of the most complex scheduling problems, is a well-known NP-hard problem of combinatorial optimization and has a very wide engineering background. In the static shop scheduling problem, finite jobs are to be processed by finite machines such that a specific optimization criterion is satisfied. According to the restrictions on the technological routes of the jobs, a general shop is indicated by a flow shop (each job is characterized by the same technological route), a job shop (each job has a specific route) and an open shop (no technological route is imposed on the jobs). The mixed job shop scheduling problem is one in which some jobs have fixed machine orders and other jobs may be processed in arbitrary orders. In past literature, optimal solutions have been proposed based on adaptations of classical solutions such as by Johnson, Thompson and Giffler among many others, by pseudopolynomial algorithms, by simulation, and by Genetic Algorithms (GA). GA based solutions have been proposed for flexible Job shops. This chapter proposes a GA algorithm for the mixed job shop scheduling problem. The chapter starts with an analysis of the characteristics of the so-called mixed shop problem. Based on those properties, a modified GA is proposed to minimize the makespan of the mixed shop schedule. In this approach, sample instances used as test data are generated under the constraints of shop scheduling problems. A comparison of the results based on benchmark data indicate that the modified GA provides an efficient solution for the mixed shop scheduling problem.

3.1 Definition of mixed –shop scheduling problem and proposed modification to the general GA algorithm

Let $J = \{J_i\}_{1 \le i \le N}$ be a set of *N* jobs to be scheduled where each job J_i consists of n_i operations. Let $O_{i,j}$ be the j^{th} operation of J_i. Let $M = \{M_k\}_{1 \le k \le m}$ be a set of *m* machines and let p_{ij} be the processing time of $O_{i,j}$ on machine $M_{i,j} \in M$. The set N is split into two subsets: $N = N_J \cup N_O$. The jobs of the set N_J have to be processed as in the job-shop; for any job $J_i \in N_J$, there is a given machine order $l_i = (M_{ik1},..., M_{ikni})$ which determines a sequence of operations of that job: $(O_{i,1},...,O_{i.n_i})$, where operation $O_{i,j}$ has to be processed after operation $O_{i,j-1}$ with j=2,…, n_i. The jobs of the set N_O have to be processed as in the open-shop; each job $J_i \in N_O$ has to be processed exactly once on each machine, and the machine order of this job is not fixed before scheduling. Given a schedule, we denote by st_{ij} and C_{ij} the starting time and completion time of operation $O_{i,j}$ ($1 \le i \le N, 1 \le j \le n_i$), respectively. The objective is to find a schedule having minimum completion time (or, *makespan*), denoted by $C_{\max} = \max_{i=1..N}(C_i)$, where $C_i = \max_{1 \le j \le n_i}(C_{i,j})$ is the completion time of job J_i [28].

An example of mixed-shop scheduling with 3 jobs and 3 machines is shown in table 1 below.

Table 1: 3 jobs by 3 machines scheduling problem

	Sequence of operations		
	1	2	3
J1 (Job Shop)	(1,3)	(2,1)	(3,2)
J3 (Job Shop)	(2,3)	(3,2)	(1,3)
J2 (Open Shop)	Any order (3,1)	Any order (1,5)	Any order (2,3)

Notation: (m,p) or (machine, processing time). For example, (2, 3) means machine 2 with processing time 3.

The following assumptions apply for the mixed-shop problem:

- Each machine can only execute one operation at a time and, once started, the operation cannot be interrupted;
- All jobs are released at time t = 0;
- There is no transportation time between machines.

Let us give each operation a task ID. Table 2 lists the jobs and operations in each job with the corresponding task ID.

Table 2: Task IDs

Job	Op.	Task ID	Job	Op.	Task ID	Job	Op.	Task ID
1	1	1	2	1	4	3	1	7
	2	2		2	5		2	8
	3	3		3	6		3	9

In the mixed shop scheduling problem, there are two kinds of constraints: precedence constraints and non-simultaneous constraints. Precedence constraints (call set ϕ) applied for jobs

that belong to the job-shop set. Non-simultaneous constraints – called set ψ - are for the set of pairs of tasks that cannot be performed simultaneously because of sequence requirement, belonging to the same job or requiring the same machine. One of the key features of this paper is the swapping technique as explained below. The modified G.A. algorithm is presented in figure 9 below.

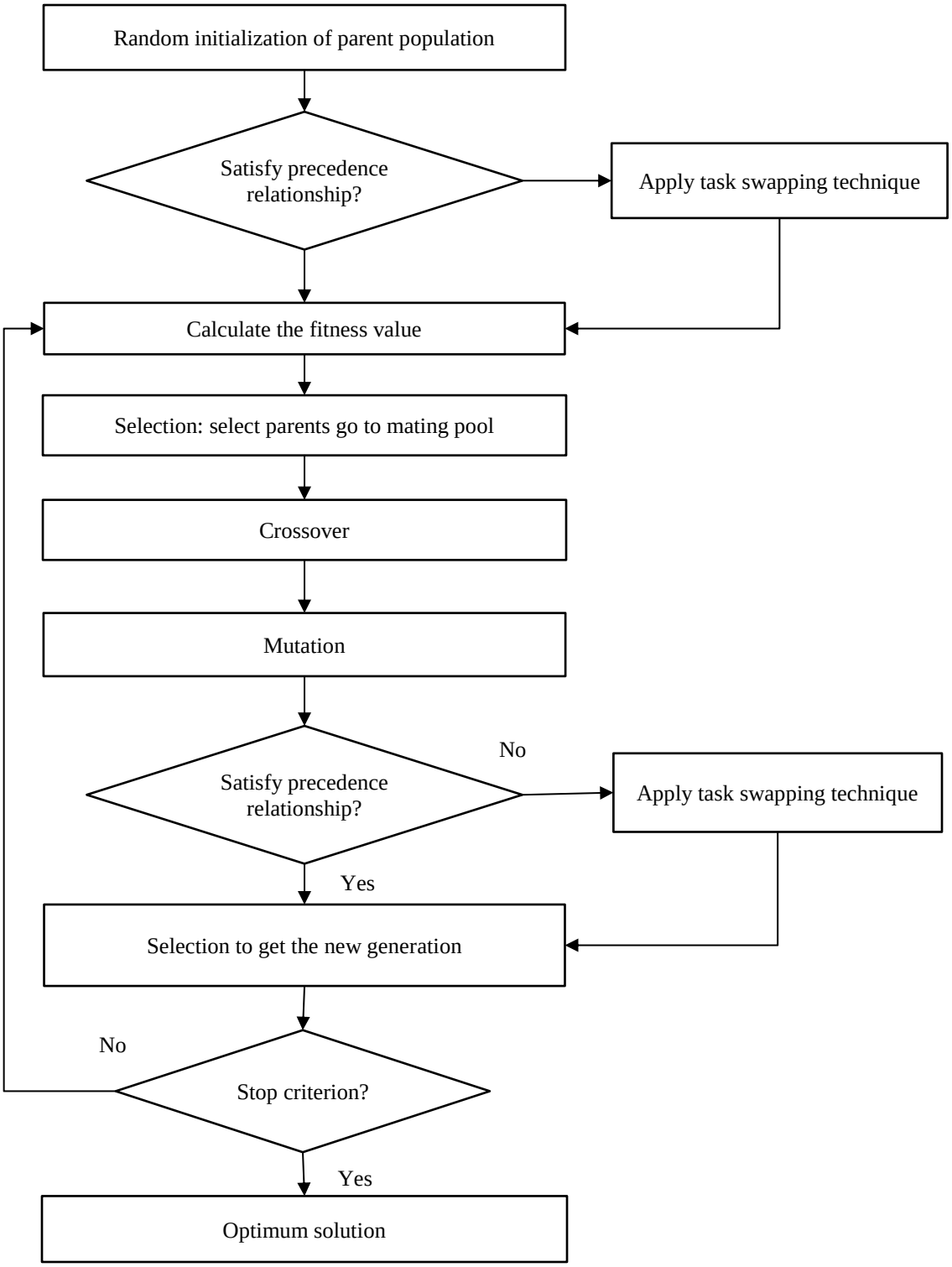

Figure 9. Modified GA flow chart

One of the key features of this paper is the swapping technique as explained in the next sections.

3.2 Chromosome representation

A chromosome of the GA represents a sequence of tasks in which a gene is a task ID.

Chromosome	1	5	3	4	9	8	6	2	7
Sequence consideration	1	2	3	4	5	6	7	8	9

Figure 10. A sample chromosome

In this chromosome, the 5^{th} gene has a value of 9; this means that the 5^{th} task (indicated in the sequence consideration) is task ID 9, which is operation 3 of job 3.

3.3 Precedence requirement

Check the chromosome to see if it satisfies the precedence constraint o. If it does not satisfy the constraint, then swap the position to make it satisfy.

In the example, $\boldsymbol{\phi}$= {(1, 2, 3), (7, 8, 9)} by observation, and there are clear violations between task IDs 3 and 2, and task ID 9 and 7. Therefore, the sample chromosome is modified as follows:

Modified chromosome	1	5	2	4	7	8	6	3	9
Sequence handling	1	2	3	4	5	6	7	8	9

Figure 11. Modified sample chromosome in figure 10

Based on the task IDs, we have the information on job, processing time and required machine as follows:

Modified chromosome	1	5	2	4	7	8	6	3	9
Sequence considering	1	2	3	4	5	6	7	8	9
Job	1	2	1	2	3	3	2	1	3
Processing time	3	5	1	1	3	2	3	2	3
Required machine	1	1	2	3	2	3	2	3	1
Operation _of job_	1 of 1	Open	2 of 1	Open	1 of 3	2 of 3	Open	3 of 1	3 of 3

Figure 12. Job, processing time required machine and operation number information corresponding to the modified chromosome in figure 11.

3.4 Objective function and fitness evaluation

Based on the sequence of operations represented by the chromosome, the objective function (makespan) can be calculated through the following procedure.

Step 1: Operation is considered based on the order in the sequence. Determine the ready time for each machine. Check type of operation.

Step 2: If operation is open shop type, then its start time is the maximum value of its machine ready time and the complete time of its non-simultaneous operations.

If operation is job shop type, its starting time is the maximum value of its machine ready time and the complete time of its precedence operation.

The complete time of the operation is the sum of start time and processing time.

Update the ready time of the machine by the completion time.

Step 3: Steps 1–2 are repeated for the next operation in sequence until the last is considered. The makespan is the maximum value of the complete time of the operation in the sequence.

For fitness evaluation, we apply a simple definition for the fitness, i.e. Fitness=1/makespan.

3.5 Tournament selection

Tournament selection provides selection pressure by holding a tournament among S competitors, with S being the tournament size. The winner of the tournament is the individual with the highest fitness of the S tournament competitors. The winner is then inserted into the mating pool.

The mating pool, being comprised of tournament winners, has a higher average fitness than the average population fitness. This fitness difference provides the selection pressure, which drives the GA to improve the fitness of each succeeding generation.

Increased selection pressure can be provided by simply increasing the tournament sizes, as the winner from a larger tournament will, on average, have a higher fitness than the winner of a smaller tournament.

3.6 Order Crossover (OX)

For order crossover and inverse mutation, we apply the following steps:

1. Given two parent chromosomes, two random crossover points are selected from one parent, partitioning it into a left, middle and right substring.

2. Produce the child chromosome by copying the middle substring of the first parent into the corresponding positions

3. Delete the tasks from the 2nd parent which have the same value with the middle substring of the first parent. The remaining tasks in the 2^{nd} parent are transferred in sequence from left to right into the empty slots of the child's chromosome.

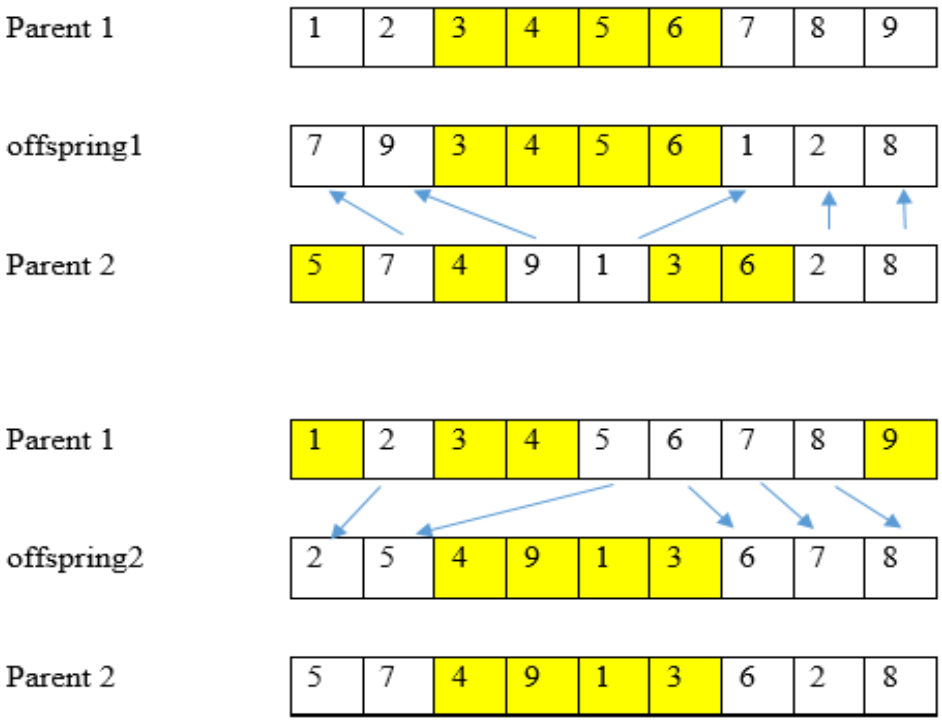

Figure 13. Illustration of OX operation

3.7 Inverse Mutation

Inverse mutation is performed on the child's chromosome by selecting 2 inverse points; then, reverse the task order in between these two points.

For example, let the selected inverse points in permutation Parent be in positions 3 and 7, and then the mutated permutation child is shown as follows.

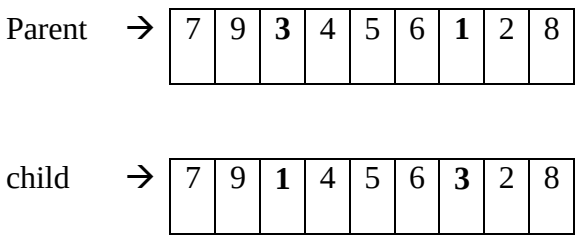

Figure 14. Illustration of inverse mutation

3.8 Selection of new generations

For the selection of new generations, we apply the Steady-State Method which consists of deleting *n* worst old members and replacing them with *n* best new members. *n* is a parameter to be experimented with.

3.9 Implementation and results

3.9.1 Optimum result for a 3 jobs x 3 machines mixed-shop

Table 3: A 3x3 sample mixed shop scheduling problem

Job	Sequence of operation (M_{ij}, P_{ij})		
	1	2	3
J1 (Job Shop)	(1,3)	(2,1)	(3,2)
J3 (Job Shop)	(2,3)	(3,2)	(1,3)
J2 (Open Shop)	Any order	Any order	Any order
	(3,1)	(1,5)	(2,3)

The results are presented in the Gantt chart in figure 15 where O_{xy} means operation y for job x. For example, O_{32} means Operation 2 for job 3

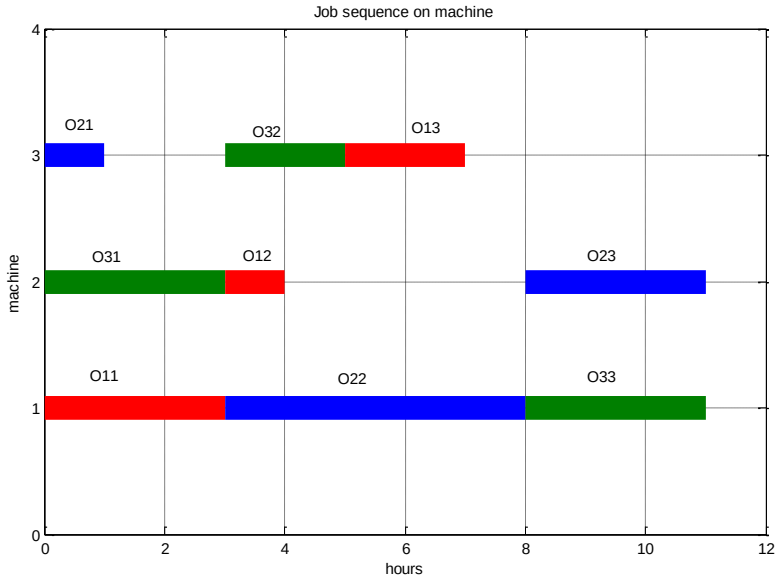

Figure 15. Gantt chart for optimal schedule 1 of mixed-shop 3 jobs 3 machines

The makespan is 11, which is equivalent to the optimum result from a Giffler Thompson solution.

3.9.2 Optimum result for a 6 jobs x 6 machines mixed-shop schedule

3.9.2.1 Benchmark problem FT06 (jobshop)

Job-shop is one special case of mixed-shop problems. FT06 is a common benchmark problem by Fisher and Thompson (1960) that has been tested by many researchers. The details of problem FT06 are described in table 4.

Table 4: The FT06 scheduling problem

Job	Sequence of operation (M_{ij}, P_{ij})					
	1	2	3	4	5	6
J1	(3,1)	(1,3)	(2,6)	(4,7)	(6,3)	(5,6)
J2	(2,8)	(3,5)	(5,10)	(6,10)	(1,10)	(4,4)
J3	(3,5)	(4,4)	(6,8)	(1,9)	(2,1)	(5,7)
J4	(2,5)	(1,5)	(3,5)	(4,3)	(5,8)	(6,9)
J5	(3,9)	(2,3)	(5,5)	(6,4)	(1,3)	(4,1)
J6	(2,3)	(4,3)	(6,9)	(1,10)	(5,4)	(3,1)

The optimal solution for the FT06 problem is 55 as obtained from the paper by Klemmt, Horn, Weigert, and Wolter [50]. They used exact methods to solve the problem.

The optimum makespan is 55, and the Gantt chart detailing the assignments of operations to machines is shown in figure 16 below.

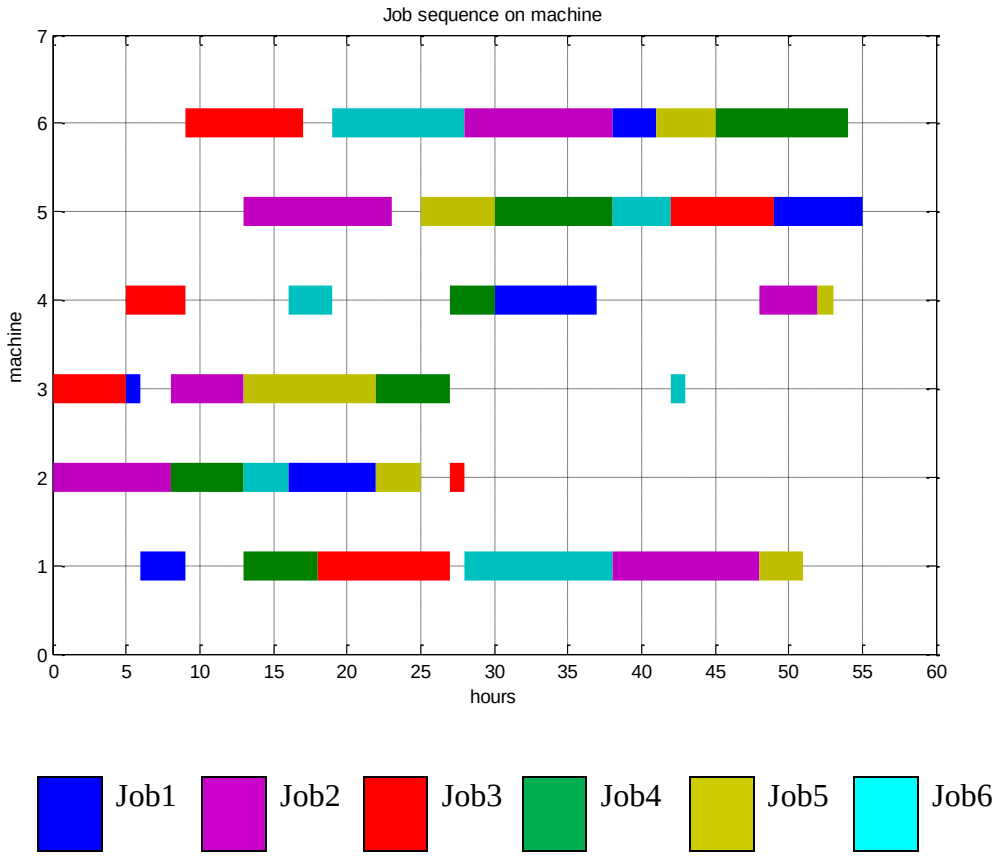

Figure 16. Gantt chart for optimal schedule of FT06 problem

Now we are going to modify this FT06 problem to make it a mixed shop problem, as indicated in table 5 below.

3.9.2.2 Modified benchmark problem FT6 (jobshop) by assuming job 2 and job 5 to be open-shop

Table 5: The modified FT06 scheduling problem

Job	Sequence of operation (M_{ij}, P_{ij})					
	1	2	3	4	5	6
J1(job shop)	(3,1)	(1,3)	(2,6)	(4,7)	(6,3)	(5,6)
J2 (open shop)	Any order (2,8)	Any order (3,5)	Any order (5,10)	Any order (6,10)	Any order (1,10)	Any order (4,4)
J3 (job shop)	(3,5)	(4,4)	(6,8)	(1,9)	(2,1)	(5,7)
J4 (job shop)	(2,5)	(1,5)	(3,5)	(4,3)	(5,8)	(6,9)
J5 (open shop)	Any order (3,9)	Any order (2,3)	Any order (5,5)	Any order (6,4)	Any order (1,3)	Any order (4,1)
J6 (job shop)	(2,3)	(4,3)	(6,9)	(1,10)	(5,4)	(3,1)

Applying our task swapping algorithm, we obtain an optimum makespan of 47. The Gantt chart indicating the assignments of operations to machine is shown in figure 17 below.

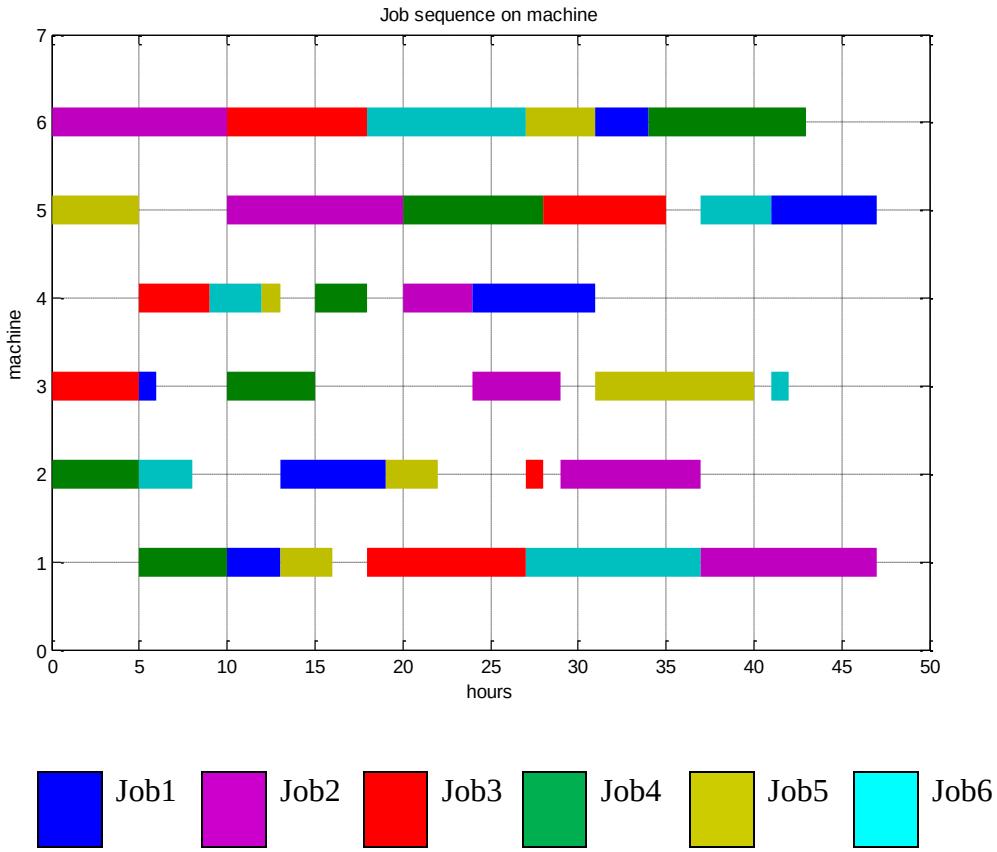

Figure 17. Gantt chart for optimal schedule of modified FT06 problem

To illustrate the effectiveness of the proposal approach, Table 6 summarizes the results based on the available bench mark problems. The bench mark problems with 3 x 3, 6 x 6, 10 x 10,15 x 15 and 20 x 20 jobs x machines are used [[48],[49]]. Note that not all bench mark problems for all three categories of job scheduling, e.g. job shop, open shop, and mixed shop, and their "optimum" solutions are available in the literature. For the cases where no optimum solution exists, the data is arbitrarily modified so that we can proceed toward solutions for all three categories of job shop, open shop and mixed shop.

Table 6: Results of proposed GA algorithm

Population Size	Job Shop			Open Shop			Mixed Shop		
	Bench Mark?	Optimum result [#]	Our result (%)	Bench Mark ?	Optimum result [#]	Our result (%)	Bench Mark ?	Optimum result [#]	Our result (%)
3 x 3	Yes	11[62]	11 (0%)	No	NA	11	No	NA	11
6 x 6	Yes	55 [63]	55 (0%)	No	NA	47	No	NA	47
10 x 10	Yes	930 [63]	960 (3.2%)	No	NA	739	No	NA	848
15 x 15				Yes	937 [64]	972 (3.7%)			
20 x 20				Yes	1155 [64]	1200 (3.8%)			

[#] Reference where optimum result was obtained

[%] Percentage difference between optimum result and our result

Looking at table 6, it is clear that our modified GA algorithm provides equivalent results with the optimum results if the population size is 6 x 6 or below. For larger populations, i.e. 10 x 10 and above, our solutions appear to be not as good as the optimum results. Nevertheless, in all cases our results are off by no more than 4%. Considering that the optimum results were obtained by analytical techniques such as Branch and Bound, and Simulated Annealing which

require substantial and complicated formulation as well as sophisticated programming skills, our straightforward modification of the GA algorithm provides a relatively easy approach and yields similar results. This feature can be considered beneficial for practicing engineers who may lack the time or the programming knowledge to write sophisticated programming code for their scheduling problems.

In the case of the 6 x 6 and 10 x 10 mixed shop problems, it is interesting to further explore the differences between our results and the ones available in [51]. Figures 18 and 19 are plots of the makespan for 10 trials and Figures 20 and 21 are plots of the processing times for 10 trials. From these figures, it is clear that the modified GA solution yields not only better and more steady results, e.g. smaller and steady make span, but also shorter processing times too.

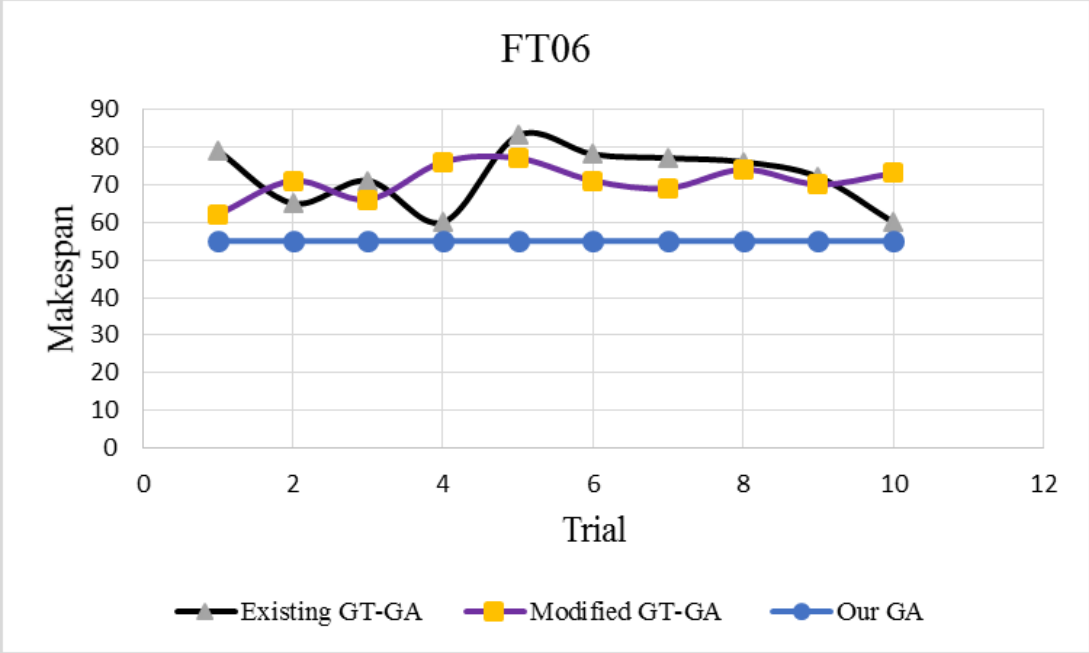

Figure 18. Comparison between the existing, modified GT-GA in [51] and our GA in makespan objective (FT 06)

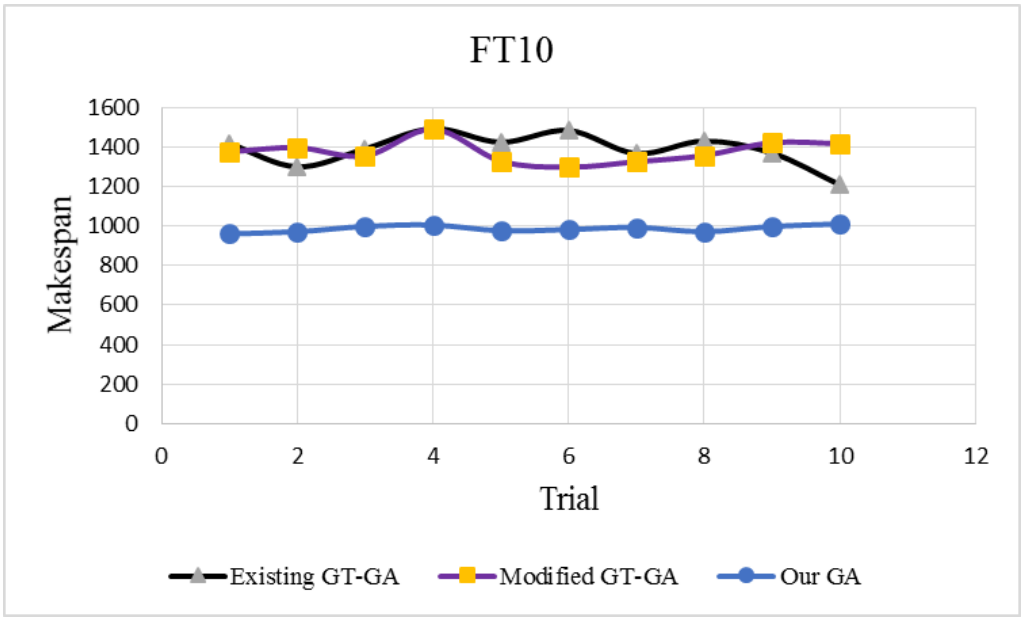

Figure 19. Comparison between the existing, modified GT-GA in [17] and our GA in makespan objective (FT10)

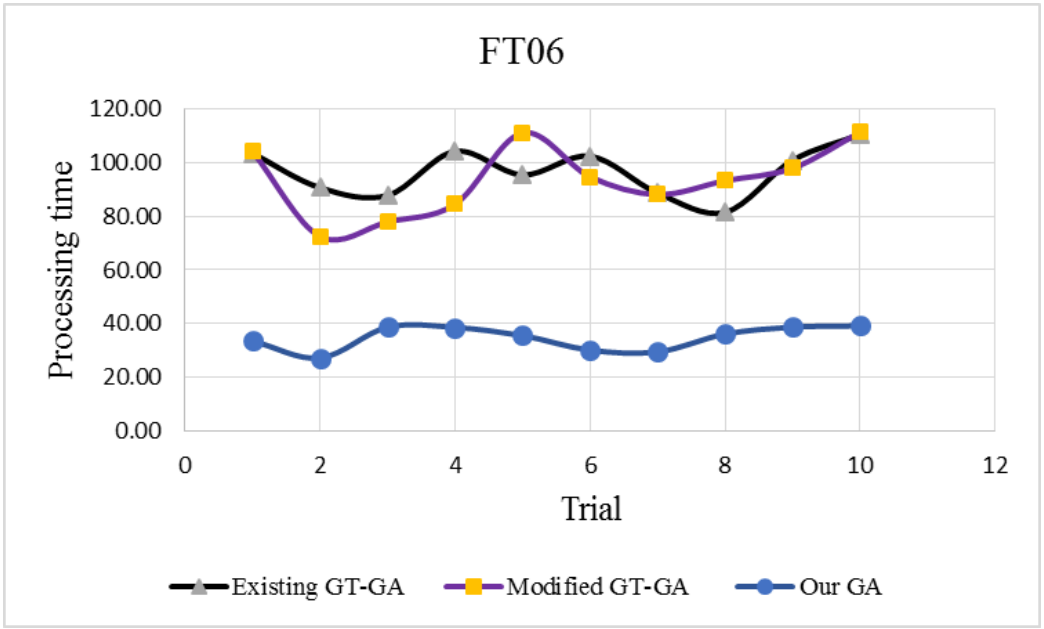

Figure 20. Comparison between the existing, modified GT-GA in [17] and our GA in processing time (FT 06)

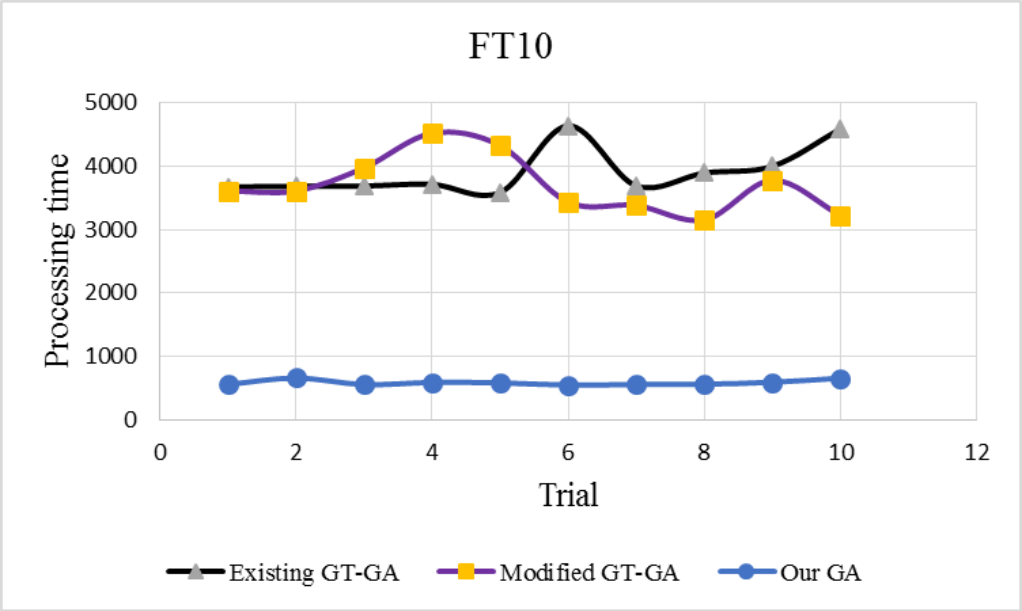

Figure 21. Comparison between the existing, modified GT-GA in [17] and our GA in processing time (FT 10)

3.10 Conclusion

The results (table 6) show that converting each operation for each job into individual task ID in combination with the classic Giffler Thompson algorithm in a general G.A. framework works very well as demonstrated in the five bench mark examples shown in table 6 above. For problem size of 6 x 6 or less, the algorithm matches the optimum solutions available in the literature, but for larger problem sizes, the results are off by no more than 4%. While the algorithm did not achieve the optimum results, it highlights the benefit of straightforward GA modification for practicing engineers who may lack the time or the programming knowledge to write sophisticated programming code for their scheduling problems. This reformulation approach together with conventional GA has allowed us to tackle the problem of machine scheduling for job shop, open shop and mixed shop requirements in an efficient manner

(computer time) and scope (type of job scheduling) (comparison results shown in Figures 18 – 21).

Although this chapter has focused on shop scheduling constrained by machines only, which is a single-resource-constrained shop scheduling problem, in a real manufacturing environment, practical SSP often needs many resources such as machine, labor, auxiliary resources (namely, maintenance equipment and tooling) each of which has a limitation, restriction or regulation that impacts shop performance. Thus, it is a multi-resources-constrained shop scheduling problem. Therefore, in the next chapter, multi-resources-constrained scheduling problem is studied with Navy ship maintenance scheduling case. The Navy ship maintenance scheduling problem was also researched with multi-mode resource constraints and multi-objective. In addition, in the next chapter, the mix-integer math formulation for the Navy ship maintenance scheduling problem is proposed. Then a mixed-integer linear program and differential evolution are applied to solve the problems to compare accurate solutions and a computation time.

CHAPTER 4

DIFFERENTIAL EVOLUTION FOR THE NAVY SHIP MAINTENANCE SCHEDULING PROBLEM

Ships play a very important role in the Navy. Naval ships are designed for combat, auxiliary duties, coastal and interdiction patrol, aircraft carriers and research purposes. Modern naval ships can be broken down into three categories: surface warships, submarines, and support and auxiliary vessels [2]. Surface warships are designed for warfare on the surface of the water and are the mainstay of naval forces. Submarines are watercraft capable of underwater operations and are designed to carry out research, rescue, or specific wartime missions. Support and auxiliary vessels are designed to support combatant ships and other naval operations such as replenishment, transport, repair, harbor, research, minesweepers, patrol, and floating hospital.

Because the quality of naval ships has a big influence on the power of combat and defense of a nation, ship maintenance must be managed and optimized carefully to increase availability, safety, reliability to increase the equipment life on ships. For any ship, maintenance activities normally must follow a maintenance cycle. A maintenance cycle represents the operation and maintenance requirements of a ship and plays an important role in keeping ships in squadrons up to date and in smooth operating condition. The activities of the maintenance cycle must be in a sequence and are usually specified for each kind of ship for different locations, weather, and operating conditions. Different ship types require different types of maintenance and different maintenance periods. The locations and availabilities of the shipyards also affect the decisions of maintenance scheduling. An efficient maintenance schedule with adequate usage of manpower and properties is necessary to maintain the ships at minimum cost.

In this chapter, the ship maintenance scheduling problem is adopted from the paper "Theory and Methodology of Ship maintenance scheduling by genetic algorithm and constraint-based reasoning" by Safaai Deris, Sigeru Omatu, Hiroshi Ohta, Lt. Cdr Shaharudin Kutar, Pathiah Abd Samat [12]. Deris, et al. state that one of the most important problems in ship maintenance scheduling is to ensure a high rate of availability. The availability and readiness of a fleet of ships determine the strength of a navy. Ship availability depends on the implementation of a Preventive Maintenance System. There is an overall desire to maintain a certain level of ship readiness year in and year out. For instance, the Royal Malaysian Navy insists on having 70% of its ships available for operations at any time. The goal of Deris' paper is to maximize ship availability while satisfying dockyard limitations. However, in Deris' paper, the number of dockyards is more than required, thus eliminating this constraint from the initial problem. The paper worked with two types of ships with different lengths of the maintenance cycles and solved for only one period cycle. To be more practical, this thesis will modify the ship maintenance cycles and limit the available dockyards. Section 4.1 will provide some concepts and a description of the optimization problem description of the ship maintenance activities. The Differential Evolution algorithm (DE) in Section 4.2 is used to search for the optimal start time of each ship maintenance activity for two case studies to maximize fleet availability. The results of numerical simulations will also be provided and analyzed to prove the reliability of the proposed algorithm. Section 4.3 will deal with the multi-mode assignment for given ship maintenance scheduling to minimize the total maintenance cost and minimize the makespan, which is the time to complete all maintenance tasks. The strategy was also developed based on DE, and the programming code and model formulation was written in Matlab. Finally, some conclusions and discussion are presented in section 4.4.

4.1 Problem Statement

The maintenance cycles utilized in both case studies investigated in this chapter consist of tasks that require a dockyard in order to be performed. Two ship classes are considered, class *A* and class *B*, and the corresponding maintenance cycles are listed in Table 7. These numbers are largely based on ship maintenance scheduling data from the Royal Malaysian Navy [12]. The time interval unit is one week. In this section, a scheduling horizon of 288 weeks is considered; this corresponds to two maintenance cycles for ships of class *A* and three maintenance cycles for ships of class *B*. A cycle of *n* weeks consists of weeks when maintenance activities are performed (*mtn*) and also weeks when the ship is available for operations (*ops*).

Table **7**: Maintenance cycle of class *A* and class *B* ships.

Class	Number of weeks											
	*mtn*1	*ops*1	*mtn*2	*ops*2	*mtn*3	*ops*3	*mtn*4	*ops*4	*mtn*5	*ops*5	*mtn*6	*ops*6
A	24	16	4	16	4	16	8	16	4	16	4	16
B	16	16	4	16	0	0	8	16	4	16	0	0

Figure 22 shows the model of maintenance scheduling of ship *p* with start and end times of each maintenance activity. As already mentioned, a maintenance cycle represents the operation and maintenance requirements of a ship over a fixed period of time. The activities of the cycle must follow the specified sequence, which is typically determined based on operational requirements, dockyard availability, and running hours of the ship. Performing maintenance activities or being available for operations during the i^{th} week implies starting from the first day of that week until the last day, e.g. if ship *p* is available for operations from the 4^{th} until the 8^{th}

week, this implies that ship p is available from the first day of the 4th week to the last day of the 8th week.

Let $st(p,i)$ be the start time of the i^{th} maintenance activity ($1 \leq i \leq I$, with I being the total number of activities during one maintenance cycle) for ship p ($1 \leq p \leq P$, with P being the total number ships in the squadron/fleet). Let also $et(p,i)$ be the end time of the i^{th} maintenance activity for ship p, $d_m(p,i)$ be the duration of the i^{th} maintenance activity, and $d_{op}(p,i)$ be the duration of the i^{th} operation activity. The formula to express the start time of the i^{th} maintenance activity of ship p in terms of the start time of the maintenance cycle $\left(st_c(p) = st(p,1)\right)$ is as follows:

$$st(p,i) = st_c(p) + \sum_{j=1}^{i-1}\{d_m(p,j) + d_{op}(p,j)\}, \text{ where } i \geq 2 \qquad (1)$$

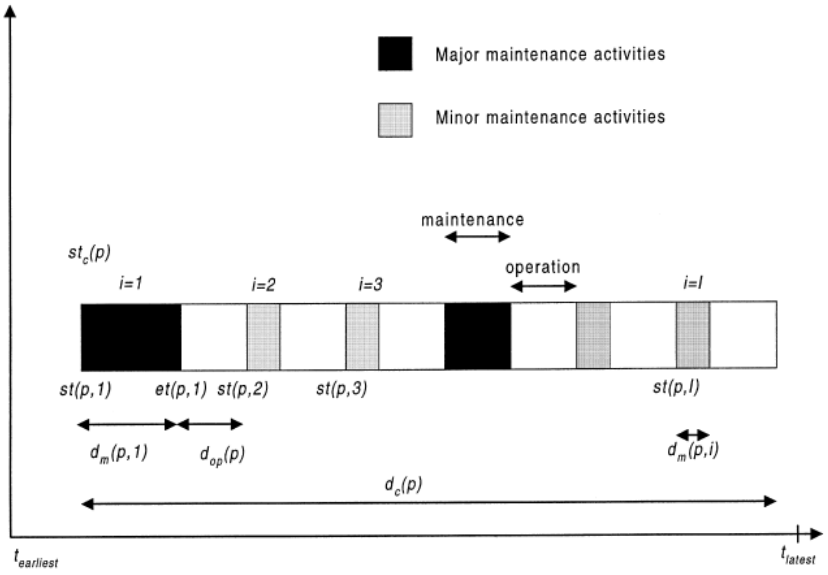

Figure 22. A maintenance cycle with precedence constraints for ship p (adapted from [12]).

The start time of the maintenance cycle, $st_c(p)$, must satisfy the following constraint:

$$t_{earliest} \leq st_c(p) \leq t_{latest} \qquad (2)$$

where t_{earliest} = 1st week and $t_{\text{latest}} = \sum_{i=1}^{I}\{d_m(p,i) + d_{op}(p,i)\}$. Considering the fact that the order in which the activities during one maintenance cycle must be performed is fixed, the decision variable corresponds to the start time of the first maintenance cycle for ship p, $st_c(p)$. Denoting the availability of ship p at any time t during the scheduling horizon as $SAv(p, st_c(p), t)$, the latter can be computed as:

$$SAv(p, st_c(p), t) = \begin{cases} 1 & et(p,i) < t < st(p,i+1) \\ 0 & st(p,i) \le t \le et(p,i) \end{cases} \quad (3)$$

The performance metrics correspond to the availability of each squadron and the availability of the fleet at time t. The availability of squadron q, $QAv(q,t)$, and the availability of a fleet consisting of Q squadrons, $FAv(t)$, can be computed as:

$$QAv(q,t) = \sum_{\substack{1 \le p \le P \\ p \in q}} SAv(st_c(p), t) \quad (4)$$

$$FAv(t) = \sum_{1 \le q \le Q} QAv(q) \quad (5)$$

where:

mtn	Weeks when maintenance activities are performed
ops	Weeks when the ship is available for operations
P	Total number of ships in the squadron/fleet
$s_t(p,i)$	Start time of the i^{th} maintenance activity for ship p
$e_t(p,i)$	End time of the i^{th} maintenance activity for ship p
$d_{op}(p,i)$	Duration of the i^{th} operation activity for ship p
$st_c(p)$	Start time of the 1st maintenance activity in each cycle for ship p
$SAv(p, st_c(p), t)$	Availability of ship p at any time t during the scheduling horizon
$QAv(q,t)$	Availability of squadron q
$FAv(t)$	Availability of a fleet consisting of Q squadrons

4.2 Differential Evolution solution for Ship maintenance scheduling

4.2.1 Optimization problem formulation for first case study, single-objective optimization

In the 1st case study, two squadrons ($Q = 2$) are considered: the 1st squadron comprises four class *A* ships and the 2nd squadron comprises eight class *B* ships, $p_q = \{4, 8\}$. The goal is to find the optimal start time of the first maintenance cycle for each ship in each squadron so that the number of weeks, during the scheduling horizon (t_f =288 weeks), with fleet availability of at least 75% is maximized.

The first constraint corresponds to the requirement that each squadron has at least 50% of its ships available during any of the 288 weeks. In addition to that, a resource constraint is imposed, which specifies that a single dockyard with five lines (*nl* = 5) is the only available resource for maintenance activities. Following the formulation used in [12], the decision-variable vector corresponds to the vector of start times, $\mathbf{st}_c$, of the first maintenance activity (*mtn*1) for each ship in a fleet of two squadrons. The corresponding optimization problem is formulated as follows:

Maximize $FAv = \sum_{1 \le t \le t_f} Fl(t)$ (6)

where:

$$Fl(t) = \begin{cases} 1, & \text{if } \left(\sum_{1 \le q \le Q} \left(\sum_{1 \le p \le p_q} SAv(st_c(p), t)\right)\right) \ge 0.75 \cdot \sum_{1 \le q \le Q} p_q \\ 0, & \text{else} \end{cases}$$

subject to:

$$\sum_{1 \le p \le p_q} SAv(st_c(p), t) \ge 0.50 \cdot p_q\,, \qquad \forall \left(q \in \{1, 2\} \wedge\ t \in \{1, 2, \ldots, t_f\}\right)$$

$$\left(\sum_{1\le q\le q_s}\sum_{1\le p\le p_q}\left(1-SAv(st_c(p),t)\right)\right)\le nl,\qquad \forall\ t\in\{1,2,\dots,t_f\}$$

4.2.2 Optimization problem formulation for second case study, Multi-objective optimization

In the 2nd case study, two squadrons with each squadron consisting of two class *A* and four class *B* ships: $p_q=\{6,6\}$ are considered. The goal is to find the optimal start times of the first maintenance cycle for each ship in each squadron so that the number of weeks, during the scheduling horizon, with squadron availability of at least 75% is maximized. Therefore, the maintenance scheduling problem becomes a multi-objective optimization problem, where each squadron competes for the available resources. The same constraints as the ones utilized in the 1st case study are also employed here. In this case though, the corresponding optimization problem is formulated as follows:

$$\text{Maximize } QAv_1=\sum_{1\le t\le t_f}Ql_1(t) \qquad (7)$$

$$\text{Maximize } QAv_2=\sum_{1\le t\le t_f}Ql_2(t)$$

where:

$$Ql_1(t)=\begin{cases}1, & \text{if } \sum_{1\le p\le p_q(1)}SAv(st_c(p),t)\ge 0.75\cdot p_q(1)\\ 0, & \text{else}\end{cases}$$

$$Ql_2(t)=\begin{cases}1, & \text{if } \sum_{1\le p\le p_q(2)}SAv(st_c(p),t)\ge 0.75\cdot p_q(2)\\ 0, & \text{else}\end{cases}$$

subject to:

$$\sum_{1\le p\le p_q}SAv(st_c(p),t)\ge 0.50\cdot p_q\,,\qquad \forall\left(q\in\{1,2\}\wedge\ t\in\{1,2,\dots,t_f\}\right)$$

$$\left(\sum_{1\le q\le q_s}\sum_{1\le p\le p_q}\left(1-SAv(st_c(p),t)\right)\right)\le nl,\qquad \forall\ t\in\{1,2,\dots,t_f\}$$

4.2.3 Differential Evolution Algorithm for Single- and Multi-Objective Optimization

The differential evolution (DE) algorithm is a population-based, stochastic optimization algorithm, which was originally developed by Storn and Price [33]. The population members, i.e., decision-variable vectors that correspond to solutions of the optimization problem, are updated through an iterative scheme that utilizes typical evolutionary operators, e.g., mutation and discrete recombination (crossover). Specifically, a randomly selected population member serves as a basis vector ($\mathbf{st}_{c,b}$) and is perturbed by a scaled difference of two other, randomly selected population members, $\mathbf{st}_{c,r1}$ and $\mathbf{st}_{c,r2}$, in order to form a trial vector ($\mathbf{st}_{c,trl}$). The decision-variable values of the trial vector can either come from the perturbed base vector or from a fourth vector, $\mathbf{st}_{c,trg}$, called the target vector. Each population member serves as a target vector once in every iteration. In the original version of DE, *DE*/*rand*/1/*bin* the number of vectors contributed by the perturbed base vector closely follows a binomial distribution (*bin*); thus, it is considered a form of uniform crossover. Furthermore, a single vector difference, i.e., 1/*bin*, is utilized to perturb the base vector and *rand* denotes random selection of individuals. More details on DE and its variants can be found in [31]. The generation of the trial vector is performed using the following scheme:

$$st_{c,trl,p}=\begin{cases}st_{c,b,p}+F\cdot\left(st_{c,r2,p}-st_{c,r1,p}\right), & \text{if } rand_p\ \le Cr\vee p=p_r\\ st_{c,trg,p}, & else\end{cases}\qquad(8)$$

$p\in\{1,2,\dots,\sum_{1\le q\le q_s}p_q\}$

where

$\mathrm{st}_{c,b}$ Basis vector

$\mathrm{st}_{c,r}$	Randomly selected population member
$\mathrm{st}_{c,trl}$	Trial vector
$\mathrm{st}_{c,trg}$	Target vector
F	The mutation scaling factor,
Cr	The crossover rate
$rand_p$	Randomly selected decision variable.

The latter is utilized is order to ensure that the trial vector differs from the target vector in at least one decision variable. The vector indices b, $r1$, $r2$, trg are mutually exclusive, i.e., $b \neq r1 \neq r2 \neq trg$. Based on the recommendations provided in [52], $F \in [0, 2]$ and $Cr \in [0, 1]$.

The trial vector is subsequently compared with the target vector by means of feasibility and objective function value. In constrained problems, such as the ones considered in this paper, the trial vector replaces the target vector in the population if any of the following conditions are true [23]:

1. The trial vector corresponds to a feasible solution while the target vector corresponds to an infeasible solution.
2. Both vectors are infeasible and the trial vector has a smaller total amount of constraint violation.
3. Both vectors are feasible and the trial vector has a better (in this case, larger) objective function value than the target vector.

It needs to be mentioned that the total amount of constraint violation is calculated as the sum of the amount of constraint violation of each of the normalized optimization problem constraints listed in the previous section. In the 2nd case study, the existence of multiple, conflicting

objectives produces a set of optimal solutions, called the Pareto-optimal set, that correspond to tradeoffs between those objectives. In this case, condition 3 is replaced by the following:

3a. If both vectors are feasible and the trial vector dominates the target vector using the following criteria for Pareto dominance (both conditions must be satisfied):

a. The trial vector is no worse than the target vector in all objectives.

b. The trial vector is better than the target vector in at least one objective.

The representation of the set of Pareto-optimal solutions in the objective-function space is termed the Pareto-optimal front. The concept of Pareto optimality is illustrated in Figure 23.

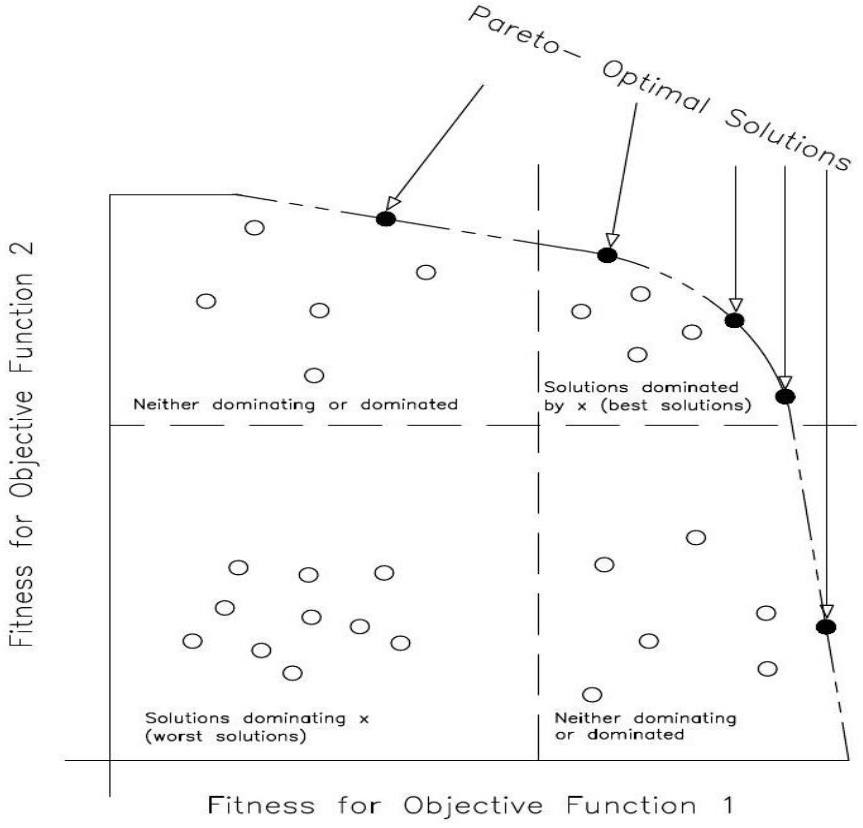

Figure 23. Concept of Pareto optimality.

4.2.4 Results and Discussion

Considering the stochastic nature of the DE algorithm, 20 runs were performed for each case study using a randomly initialized population of 120 solution vectors; the initialization of each run was based on a different random seed. The values of the parameters *F* and *Cr* were set equal to 0.5 and 0.1, respectively, and were kept fixed throughout each run. A small value of *Cr* results in trial vectors with decision-variable values mainly obtained from the target vector and,, thus, promotes exploitation of the search space region near the current position of the target vector. Through experimentation with different values of *F*, it was determined that the algorithmic performance is not affected significantly by the choice of its value; therefore, a value of 0.5 was utilized in all runs. The number of iterations per run was set equal to 1,000, which corresponds to 120,000 problem evaluations, and was utilized as the termination criterion. The same parameters were used for both case studies.

4.2.4.1 First case study

For the 1st case study, the optimization problem formulated in Eq. (6) was solved and the 10 best solutions obtained, all of which are feasible, and are listed in Table 8. The global optimal solution that was found is the solution listed as solution no. 5 in Table 8. This solution provides the highest number of weeks with fleet availability of at least 75%. Specifically, it provides 88 weeks with 75% fleet availability (9 ships) and 12 weeks with 83.3% availability (10 ships). The results listed in Table 2 also reveal that there exist many local optima for this particular optimization problem, nevertheless, the optimal start time of *mtn*1 of certain ships, e.g., ships # 3, # 6, and # 7 in the 2nd squadron seems to be the same regardless of the start time values of *mtn*1 of the other ships in the squadron. The maintenance schedule of the two squadrons based

on solution no. 5 is shown in Figs. 24 and 25, respectively. The corresponding availability charts are plotted in Figs. 26 and 27.

The fact that the objective function formulation is not biased towards any specific value of the fleet availability over 75% allowed the algorithm to find a diverse set of solutions. For instance, solution no. 3 provides two weeks with fleet availability of 11 ships. This is due to the fact that the maintenance schedule of the 1st squadron based on solution no. 3 provides several weeks with squadron availability of 100% (4 ships), as shown in Figure 28.

Table 8: Computed optimal solutions for the 1st case study – optimal start time of $mtn1$ for each ship.

Ship # / Solution #	1st Squadron				2nd Squadron								Fleet availability			
													# of weeks			
	1	*2*	*3*	*4*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	with 9 ships	with 10 ships	with 11 ships	FAv
1	65	1	16	52	65	69	1	17	49	33	81	53	80	16	0	96
2	65	1	15	51	65	68	1	17	49	33	81	52	79	16	0	95
3	85	69	36	21	65	9	1	16	45	32	81	57	83	8	2	93
4	89	105	41	57	69	77	1	17	29	37	81	53	80	16	0	96
5	89	1	37	53	65	25	1	21	49	29	81	37	88	12	0	100
6	89	1	37	53	65	67	1	77	51	29	81	35	82	14	0	96
7	88	100	36	52	65	9	1	17	49	33	81	45	81	13	0	94
8	94	80	30	44	65	35	1	17	48	32	81	51	78	16	0	94
9	121	65	13	29	61	45	1	14	49	29	37	50	85	8	0	93
10	121	105	13	25	65	65	1	17	49	33	81	29	84	12	0	96

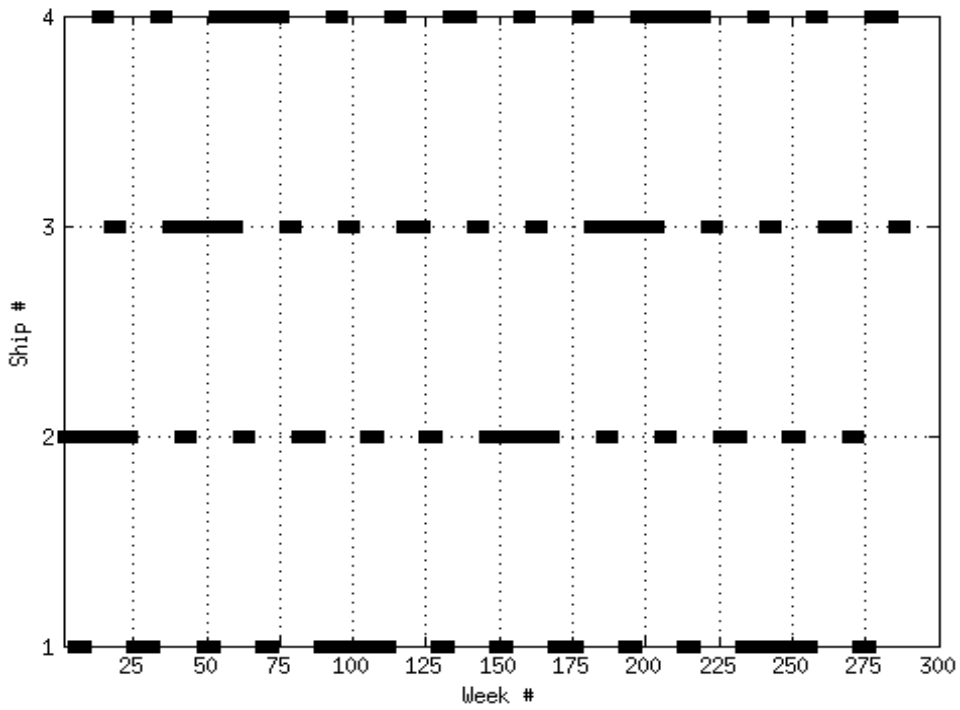

Figure 24. Optimal maintenance schedule of the 1st squadron (solution no. 5 in Table 8).

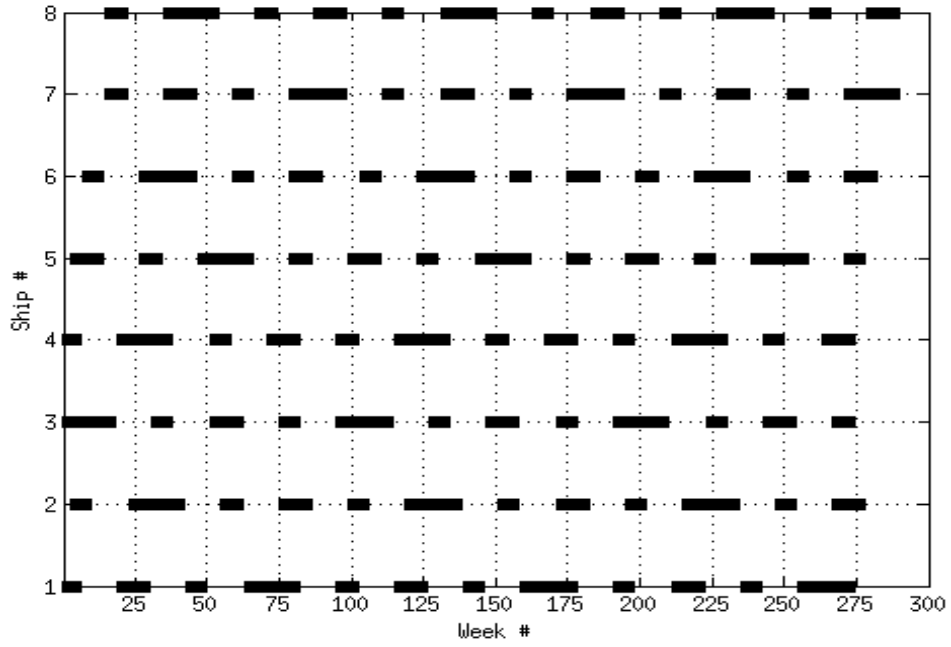

Figure 25. Optimal maintenance schedule of the 2nd squadron (solution no. 5 in Table 8).

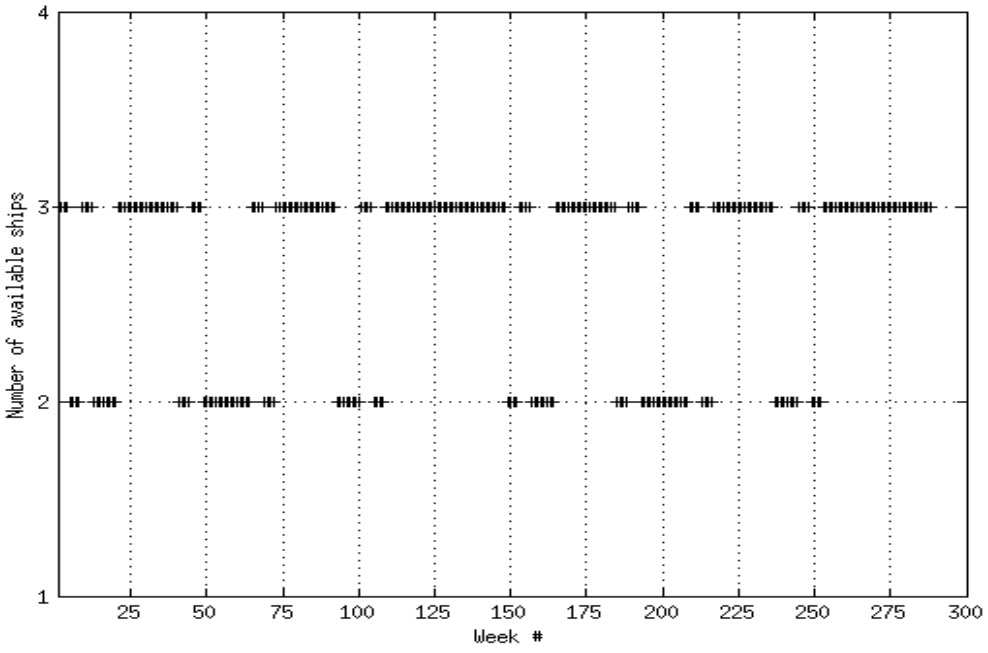

Figure 26. Availability of the 1st squadron using solution no. 5 (from Table 8).

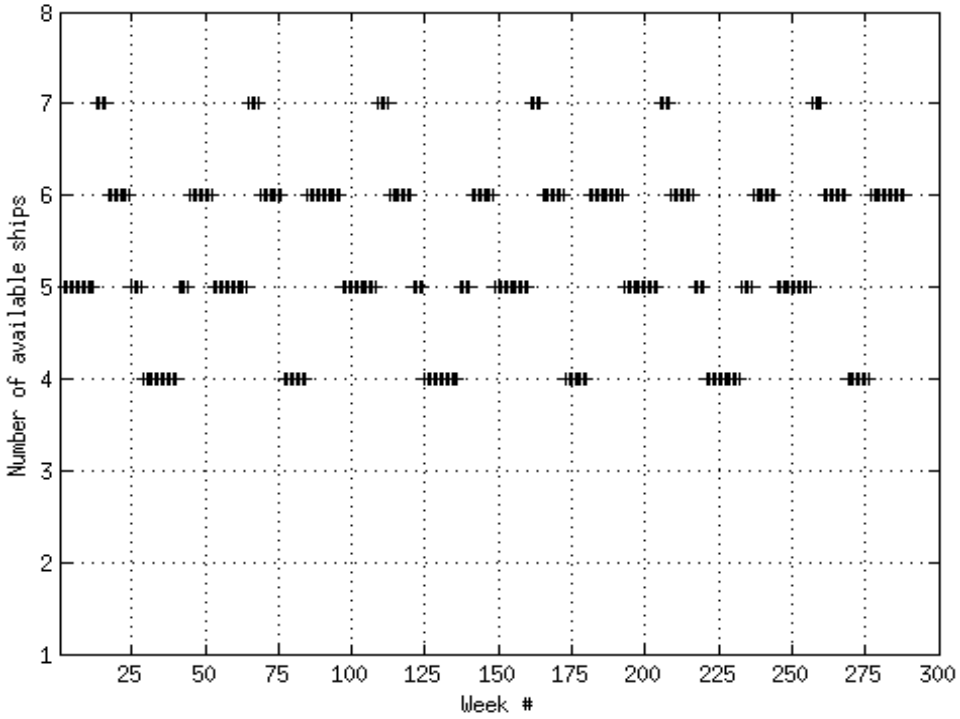

Figure 27. Availability of the 2nd squadron using solution no. 5 (from Table 8).

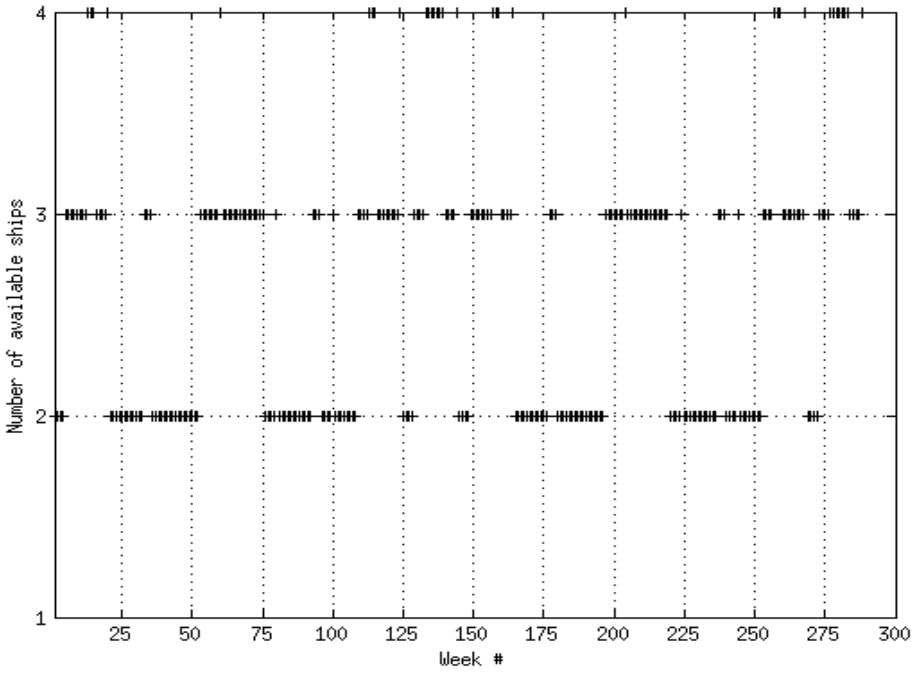

Figure 28. Availability of the 1st squadron using solution no. 3.

4.2.4.2 Second case study

In the 2nd case study, two squadrons, each consisting of two class *A* and four class *B* ships, are competing for maintenance resources. This is a typical situation of destroyer squadrons consisting of destroyers and frigates of various classes. The multi-objective problem formulation presented in Eq. (7) was solved using the DE algorithm.

- The computed Pareto-optimal front is shown in Figure 29. It can be observed that the computed solutions correspond to tradeoffs between objectives, i.e., increasing the number of weeks with availability of at least 75% of one squadron results in decreasing the number of weeks with availability of at least 75% of the other. It needs to be noted that the Pareto-optimal front allows the decision maker to select a maintenance schedule that fits their preferences based on the available solutions and fleet operating requirements.

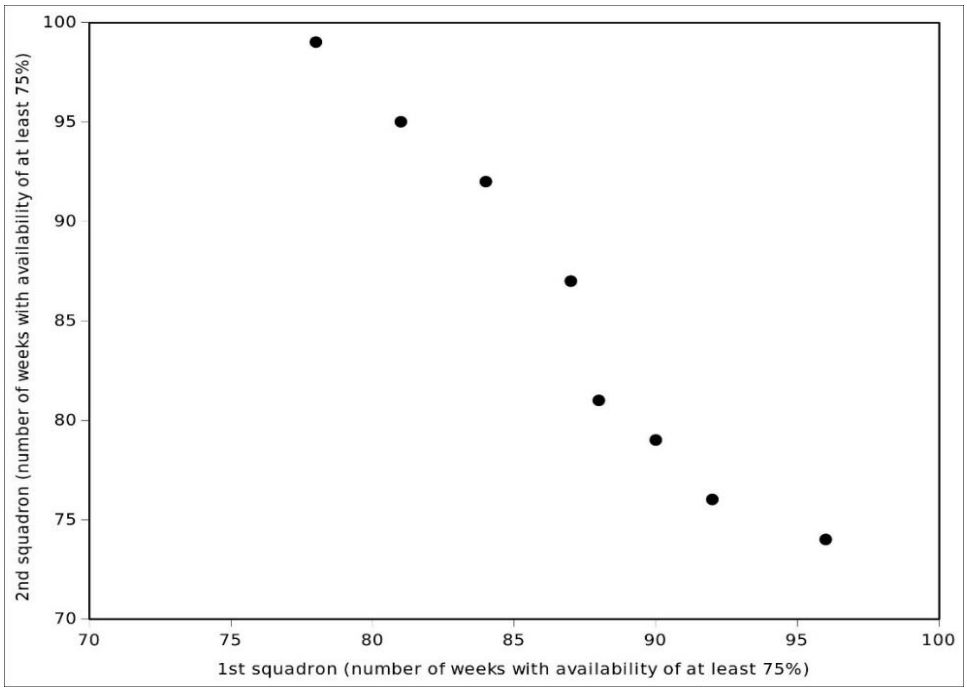

Figure 29. Pareto-optimal front for number of weeks with squadron availability of at least 75%.

In this session, variants of the ship maintenance scheduling optimization problem were formulated and solved using the differential evolution (DE) algorithm. In the first case study, the DE optimizer was able to find a unique global optimal solution and also, being a population-based algorithm, provide a set of suboptimal, but diverse solutions, which could potentially be viable alternatives if the fleet availability requirements changed during the considered scheduling horizon. In the second case study, it was shown that the DE algorithm can successfully solve the ship maintenance scheduling optimization problem when this is formulated as a multi-objective problem, e.g., when multiple squadrons 'compete' for the same dockyard resources. In this way, it allows the decision maker to consider tradeoffs between the conflicting objectives some posteriori and select the solution that best matches their preferences.

4.3 Differential Evolution for the multi-mode resource-constrained ship scheduling

problem

In section 4.2, Pareto Differential Evolution (PDE) is employed to create new solutions. And the results showed the competence of PDE for solving multi-objective optimization problems (MOPs) in scheduling field. In this section, the studies are continuously concentrated on DE for MOPs, with the multi-mode resource-constraints. In the multi-mode resource-constrained ship scheduling problem (MRCSSP), the ship maintenance schedule is given. However, now each maintenance activity has a set of execution modes with different duration time-resource alternatives. Each type of resource has given cost rate and a quantity limitation per week. This problem aims at minimizing the total duration or makespan of a maintenance schedule and total maintenance cost subject to precedence relations between the maintenance activities, and at least 50% of ships must be available per week subject to limited renewable resource availabilities.

4.3.1 Problem statement

We adopt the maintenance cycle of the four ships belonging to class A from the paper "Ship maintenance scheduling by genetic algorithm and constraint-based reasoning" by Safaai Deris, Sigeru Omatu, Hiroshi Ohta, Lt. Cdr Shaharudin Kutar, Pathiah Abd Samat [12] as follow:

Table 9: Maintenance cycle for ships in class *A*

Class	Number of weeks											
	*mtn*1 major	*ops*1	*mtn*2 minnor	*ops*2	*mtn*3 minor	*ops*3	*mtn*4 mid	*ops*4	*mtn*5 minor	*ops*5	*mtn*6 minor	*ops*6
A	24	16	4	16	4	16	10	16	4	16	4	16

The maintenance schedule is given. For example, figure 30 is a given maintenance scheduling for four ships class A in which the start times of the major maintenances of the four ships are respectively on week 123th, week 92th, week 1st, and week 32th

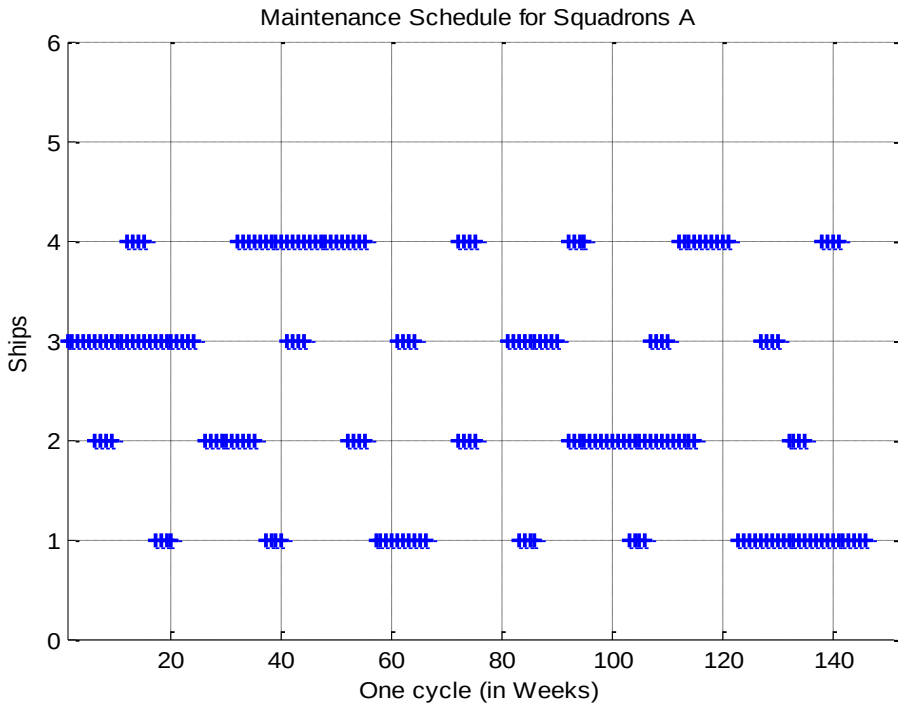

Figure 30. Given maintenance schedule for four ships class A

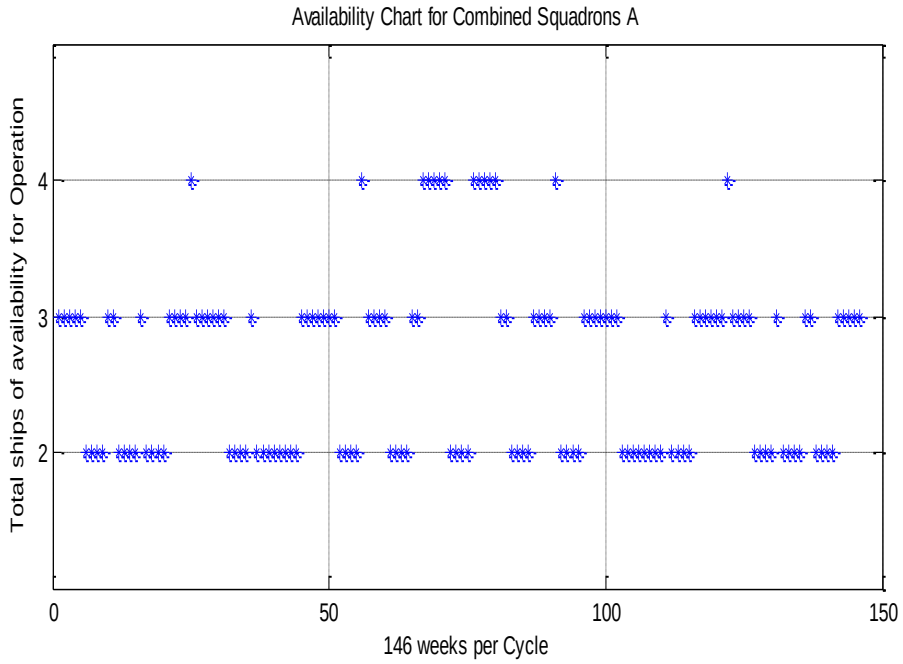

Figure 31. Total ships class A is available for operation on specific week

There are three types of maintenance activities: major, mid-range and minor. Each type of ship maintenance activity can be executed in various modes. Each mode affects the activity's duration, resource requirements and cost. The data is given as follows, where S1, S2, and S3 are respectively the three types of resources. Now the problem is expanded with introduction of 3 modes of operation shown in table below:

Table 10: Different modes of maintenance type

Type of maintenance	Mode 1	Mode 2	Mode 3
Major	24 weeks using	20weeks using	18 weeks using
	20 of S1	20 of S1	20 of S1
	10 of S2	15 of S2	20 of S2
	5 of S3	10 of S3	15 of S3
Mid-range	10 weeks using	8 weeks using	6 weeks using
	10 of S1	10 of S1	10 of S1
	5 of S2	10 of S2	15 of S2
	5 of S3	10 of S3	15 of S3
Minor	4 weeks using	3 weeks using	2 weeks using
	5 of S1	5 of S1	5 of S1
	2 of S2	5 of S2	10 of S2
	2 of S3	5 of S3	10 of S3

The cost rates for the resources are given as follows:

Table 11: Rate for resources

Resource type	Cost per week
S1	$1,000
S2	$1,500
S3	$2,000

The resources have limitations per week:

Table 12: Resource limitation, per week.

Resource	Number
S1	35
S2	30
S3	20

The objective of the MRCSSP is to schedule maintenance activities to minimize the total duration or makespan of a maintenance schedule and the total maintenance cost, subject to the fixed sequence of maintenance types, the specific quantity of possibly several renewable constrained resources, and the operating availability constraints. In next paragraph, a mix-integer linear math model for MRCSSP is formulated. Then DE will be analyzed to show its advantages to solve the MRCSSP.

4.3.2 Math formulation for multi-mode resource-constrained ship scheduling problem

Notation:

t	Type of maintenance activity
m	Mode
d_{tm}	Maintenance duration time type t mode m
$S1_{tm}, S2_{tm}, S2_{tm}$	Resources S1, S2, S3 type t mode m
$C1, C2, C3$	Cost for resources S1, S2, S3

$d_{ma}(p,i,m,t)$ Maintenance duration time of ship p, activity i, type t, mode m

$d_{op}(p,i)$ The operation duration time of ship p activity i

$st_c(p)$ Start time of first maintenance activity of ship p

$st(p,i)$ Start time of maintenance activity i of ship p

$S1(p,w)$ Required resources S1 for ship P at week w

$S2(p,w)$ Required resources S2 for ship p at week w

$S3(p,w)$ Required resources S3 for ship p at week w

$Makespan(p)$ Required time to complete one cycle of ship p

$Makespan$ Required time to complete all ship actives of a squadron

T The maximum planning makespan

M A large number, say M= 10000

Variables:

$$x_{pim} = \begin{cases} 1 \; if \; activity \; i \; of \; ship \; p \; has \; been \; assigned \; mode \; m \\ 0 \; otherwise \end{cases}$$

$$S_{pimw} = \begin{cases} 1 \; if \; activity \; i \; of \; ship \; p \; is \; maintain \; on \; mode \; m \; at \; week \; w \\ 0 \qquad\qquad otherwise \end{cases}$$

Objective functions: minimize the makespan (Makespan) and minimize the total maintenance cost (Total cost)

Makespan= max(makespan(p)), Where p = 1 to q

$$Makespan(p) = \sum_{i=1}^{I} \sum_{m=1}^{3} \left(x_{pim} * d_{ma}(p,i,m,t) \right) + 96$$

96 is the total operation time per cycle

$$\boldsymbol{Total\ cost} = \sum_{p=1}^{q}\left(\sum_{i=1}^{I}\sum_{m=1}^{3} x_{pim} * d_{ma}(p,i,m,t) * S1_{tm} * C1\right.$$
$$+ \sum_{i=1}^{I}\sum_{m=1}^{3} x_{pim} * d_{ma}(p,i,m,t) * S2_{tm} * C2$$
$$\left. + \sum_{i=1}^{I}\sum_{m=1}^{3} x_{pim} * d_{ma}(p,i,m,t) * S3_{tm} * C3\right)$$

Subject to:

$\sum_{m=1}^{3} x_{pim} = 1$ (1)

$\sum_{w=1}^{T} S_{pimw} = x_{pim} * d_{ma}(p,i,m,t)$ (2)

$st(p,i) \leq S_{pimw} * w + M(1 - S_{pimw})$ (3)

$et(p,i) \geq S_{pimw} * w$ (4)

$st(p,1) = st_c(p)$ (5)

$st(p,i) = st_c(p) + \sum_{j=1}^{i-1}\{\sum_{m=1}^{3}\left(x_{pjm} * d_{ma}(p,j,m,t)\right) + d_{op}(p,j)\}$,

where $i \geq 2$ (6)

$et(p,i) = st(p,i) + \sum_{m=1}^{3}\left(x_{pim} * d_{ma}(p,i,m,t)\right)$ (7)

$\sum_{p=1}^{q}\sum_{m=1}^{3} S_{pimw} * S1_{tm} \leq 35$ (8)

$\sum_{p=1}^{q}\sum_{m=1}^{3} S_{pimw} * S2_{tm} \leq 30$ (9)

$\sum_{p=1}^{q}\sum_{m=1}^{3} S_{pimw} * S3_{tm} \leq 20$ (10)

$\sum_{p=1}^{q}\sum_{m=1}^{3} S_{pimw} \leq 0.5 * q$ (11)

Constraint (1) makes sure that each activity can only be processed in one mode.

Constraint (2) shows the relationship between S_{pimw} and x_{pim}. If activity i of ship p is assigned mode m, i.e x_{pim} =1, then the total maintenance week of activity i on mode m is equal to the duration maintenance time of mode m type t (t is type of maintenance activity i-known). If activity i of ship p is not assigned for mode m, then x_{pim}=0, and the total maintenance week of activity i on mode m is 0.

Constraint (3) and Constraint (4) force S_{pimw} =1 if $st(p,i) \le w \le et(p,i)$

Constraint (5) forces the start time of the first maintenance activity of ship p to be equal to the start time of the maintenance cycle, known

Constraint (6) and Constraint (7) are used to calculate start time and end time of activities in the sequence.

Constraint (8), (9), (10) are limitations of resource S1, S2, and S3 at week w, where w=1,…,T

Constraint (11) guarantees that each week has at least 50% of all ships available for operation.

Although, the problem is a mixed integer linear problem, the large number of variables and constraints make the problem a very difficult problem to solve. For example, let's consider a small-size problem with only 4 ships. Each ship has six maintenance activities. Each type of maintenance activity has three different modes. The length of the maintenance cycle is 146 weeks. The total number of variables for this problem is therefore 10584 variables. Attempting to develop a solution using the mixed integer linear model is prohibitive due to the large number of variables. Therefore, this attempt was abandoned in favor of a DE solution as explained in the section below.

4.3.3 Differential Evolution Algorithm for the multi-mode resource-constrained ship scheduling

problem

This part will introduce two methods of calculating the fitness value used in the DE algorithm to solve MRCSSP. The first method, which is the simplest and most widely-used method, is the weighted sum method. The weighted sum method transforms multiple objectives into an aggregated objective function by multiplying each objective function by a user supplied weight and summing up all weighted objective functions. The multi-objective problem (MOP) then becomes a single-objective optimization and the decision for DE to create a new generation of feasible solutions is easy: the candidate replaces the parent only when the candidate is better than the parent.

The second method is Pareto Differential Evolution (PDE). PDE will use the concept of dominance of pareto selection to create the new population: the candidate replaces the parent if it dominates its parent or if both candidate and parent are non-dominated regarding each other. If the parent dominates the candidate, the candidate is discarded. This step is repeated until the targeted population size is reached.

4.3.3.1 Weighted Sum method

In MOPs, different objectives can take different orders of magnitude or be calculated with different units. For example, the total cost of maintenance activities may vary between 80,000 dollars to 100,000 dollars while the makespan may vary from 130 days to 146 days. When such objectives are weighted to form a composite objective function, normalization the objectives is necessary which scale objectives appropriately so that each objective can be assessed equivalently with respect to any other objective.

The Maintenance scheduling utility, MSU, is introduced as the objective function for MRCSSP. Optimal mode assignment means maximizing average availability percentage,

minimizing the average makespan, and minimizing total cost. The formulation of MSU is shown as:

$$MSU = w_1[U_a] + w_2[U_m] + w_3[U_c]$$

Where: U_a is utility of ship availability, U_m is utility of makespan, U_c is utility of cost, and w_j is the weight respectively of availability, makespan and cost.

Although the weighted sum method simple and easy to use, finding the optimal weighted value is non-simple. It depends on the relative importance of each objective and a scaling factor which we will address later. In this section, we assume that based on customer's preferences, the weight values as follows:

	Weight
Availability	0.45
Makespan	0.20
Cost	0.35

Therefore, the objective function is

$$MSU = 0.45[U_a] + 0.2[U_m] + 0.35[U_c]$$

In the conventional utility theory [53],

$$U = \frac{1 - e^{-rs}}{1 - e^{-r}}$$

With:

$$s = \frac{x - x_{worse}}{x_{best} - x_{worse}}$$

Where:

r is a measure of the risk attitude of the customer

s is a normalized form of the outcome x

In this research, we apply the risk neutral approach meaning that r=0. Applying L'Hospital's rule [54], this leads to U = s

$$U_a = s_a = \frac{Average\ availability - worst\ availability}{Best\ availability - worst\ availability}$$

$$U_m = s_m = \frac{Average\ makespan - worst\ makespan}{Best\ makespan - worst\ makespan}$$

$$U_c = s_c = \frac{Average\ total\ cost - worst\ total\ cost}{Best\ total\ cost - worst\ total\ cost}$$

A determination of best and worst case scenarios reveal that:

Best Availability = 75%

Worst Availability = 65.75%

Best Makespan = 128 weeks

Worst Makespan = 146 weeks

Best Total cost =4x $ 1547000=6188000

Worst Total cost =4x $ 2135000=8540000

The constraints are:

$$\sum_{i=1}^{4} S1_{ik} \leq 35$$

$$\sum_{i=1}^{4} S2_{ik} \leq 30$$

$$\sum_{i=1}^{4} S3_{ik} \leq 20$$

$$\sum_{i=1}^{4} p_{ik} \leq 0.5 * q$$

Where k is the week number k.

Differential Evolution for MRCSSP:

A mode vector represents a solution. The length of the vector is equal to the number of ships multiplied by the number of maintenance activities for each ship for example, there are 4 ships and each ship has six maintenance activities, then the length of the mode vector is 4x6=24 where 4 represents the 4 ships, and 6 represents the 6 maintenances required in a maintenance cycle.

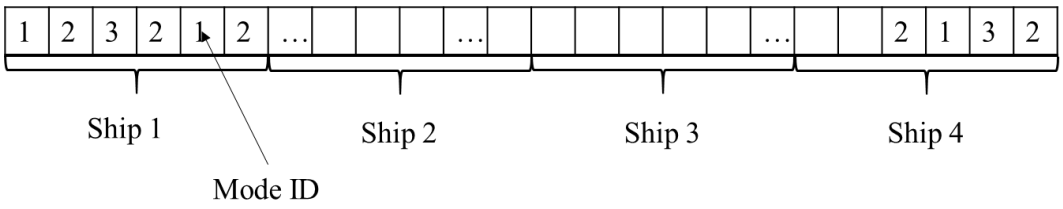

Figure 32. Mode vector for MRCSSP

The values in locations 1 to 6 of vector represent the mode assignments for ship 1

The value in locations 7 to 12 of vector represent the mode assignments for ship 2

The value in locations 13 to 18 of vector represent the mode assignments for ship 3

The value in locations 19 to 24 of vector represent the mode assignments for ship 4

Pseudo code:

While stopping conditions not true **do**

for each target vector, $x_{i,G}$ (where G is the generation number, i=1, …, NP), $x_{i,G} \in C(G)$ **do**

Evaluate the fitness values $f_1(x_i)$**,** $f_2(x_i)$**,** and $f_3(x_i)$**,** and check the constraint satisfaction. If it is violated the constraints, give the fitness value a penalty

Randomly select three vectors $x_{r1,G}, x_{r2,G}, x_{r3,G}$ x such that the indices i, r1, r2 and r3 are distinct

Generate scaling vector $V_i : V_{i,G+1} = x_{r1,G} + F\left(x_{r2,G} - x_{r3,G}\right)$

Check boundary for all V_i to ensure it within the range

Recombine each target vector $x_{i,G}$ with scaling vector V_i to generate a trial vector U_i using equation:

$$U_{j,i,G+1} = \begin{cases} V_{j,i,G+1} \; if \; rand_j \le C_r \\ x_{j,i,G} \quad Otherwise \end{cases}$$

Check whether each variable of the trial vector is within range and make it within range

Calculate the objective function value for trial vector U_i.

Choose better of the two (function value at target and trial point) for next generation C(G+1) using equation:

$$x_{i,G+1} = \begin{cases} U_{i,G+1} \; if \; f\left(U_{i,G+1}\right) \le f\left(x_{i,G}\right) \\ x_{i,G} \quad otherwise \end{cases}$$

end

end

Parameter selections:

The selection of the best f value and Cr value is based on the Golden Section Search Method [55] as explain below:

First, a random Cr value is chosen. Then MSU results for various F value are obtained as shown in table below:

Table 13: MSU results for fixed Cr equal to 0.9

F	0	0.1	0.05	0.15	0.075	0.125	0.9
MSU	0.5528	0.6726	0.6648	0.6648	0.6593	0.6593	0.6593

Second, the best result from table 13, i.e F=0.1, is selected and the MSU results are obtained for various Cr values as shown in table 14:

Table 14: MSU results for fixed F equal 0.1

Cr	0.8	0.7	0.9	0.85	0.5	0.1
MSU	0.6648	0.6593	0.6726	0.6648	0.6648	0.6593

The previous 2-steps process in repeated until no further improvement in MSU is observed. In conclusion, the best results for MRCSSP are obtained when Cr=0.9 and F=0.1.

Constraint-Handling techniques:

There are a number of efficient methods for handling constraints. In this thesis, three methods are tested for their stability and robustness for MRCSSP. The first proposed algorithm is a differential evolution (DE) algorithm using death penalty functions for constraint handling. The second method is penalty functions. The third algorithm is using Deb's constraint handling method.

1. Death penalty:

When a certain solution violates a constraint, it is considered as meeting the death penalty and is rejected and generated again. Thus, no further calculations are necessary to estimate the

degree of infeasibility of such a solution. Table 15 shows the results of 5 runs using the death penalty constraint-handling technique.

Table 15: Optimal results of 5 runs DE with death penalty handling constraint method

Run	MSU
1st	0.622
2nd	0.6087
3rd	0.6141
4th	0.622
5th	0.6044

Note: with death penalty, results are not stable or uniform.

Table 16 shows the best results for scheduling the maintenance of 4 ships.

Table 16: Best solutions of DE with death penalty handling constraint method

Input: start time of ship major maintenances				Opt MSU value	Number of weeks in violation	Average Makespan	Total cost	Average availability
Ship 1	Ship 2	Ship 3	Ship 4		(>50% ship maintain/week)	(days)	(million USD)	
119	7	103	69	0.622	0	132.5	7.478	72.4528302
Mode assignment with same optimal MSU value								
1st mode assignment: 1 3 1 3 3 2 1 3 3 3 3 3 3 3 3 3 2 3 3 3 3 2 3 3								
2nd mode assignment: 3 1 1 3 3 2 1 3 3 3 3 3 3 3 3 3 2 3 3 3 3 2 3 3								

Figure 33 shows a Gantt chart for the maintenance schedule of the 4 ships. The following explanation is provided to help the understanding of figure 12. The mode assignment is indicated as 1, 3, 1, 3, 3, 2, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 2, 3, 3, 3, 3, 2, 3, 3. this means that the 1st 6 digits, i.e. 1, 3, 1, 3, 3, 2 are the respective modes fir ship 1's series of maintenances shown on the figure 12 for ship 1, i.e. minor, minor, mid-range, minor, minor, and major maintenance.

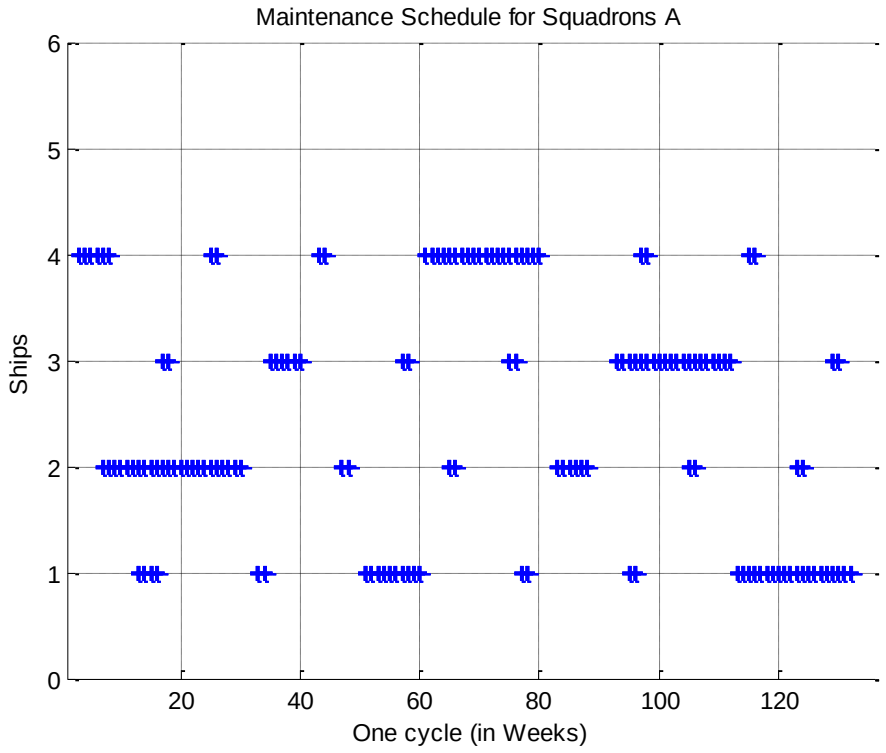

Figure 33. Graph correspond with 1st mode assignment in table 16

Figure 34 shows the 2rd mode assignments for the same 4 ships.

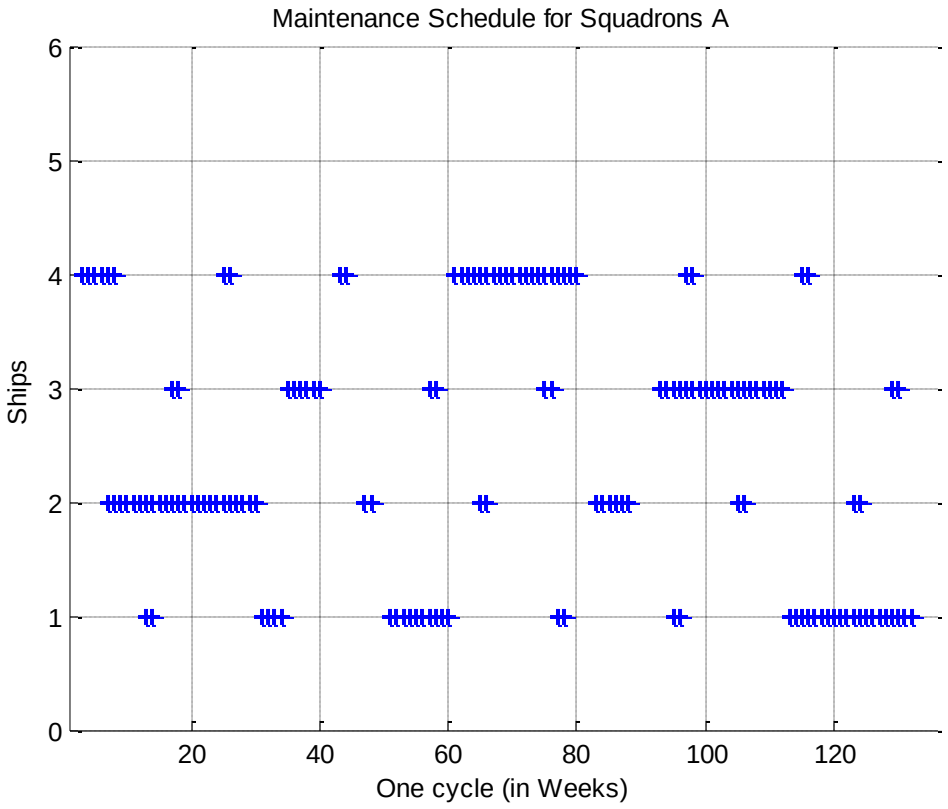

Figure 34. Graph correspond with 2nd mode assignment in table 16

2. Penalty function

We handled the volume constraints by adding a penalty proportional to the constraints violation to the objective function value. Penalty function is formulated based on the number of constraints violated:

Fitness= MSU - R_j* Penalty

Where:

$$Penalty = max\left(0, \sum_{i=1}^{C_{max}} V_i\right)$$

$$V_i = \begin{cases} 0 \; if \; at \; week \; i \; all \; constraints \; are \; sastisfy \\ 1 \; if \; at \; week \; i \; at \; least \; one \; constraint \; is \; violated \end{cases}$$

$\mathbf{R_j}$ = 10, a chosen quantity.

Table 17 show the optimal results of five DE runs using the penalty function constraint handling method.

Table 17: optimal results of 5 DE runs with penalty function constraint handling method

Run	MSU
1st	0.622
2nd	0.622
3rd	0.622
4th	0.622
5th	0.622

It is noted that the MSU results are uniform for all 5 runs. Using the data for the first run the optimal modes are 3, 1, 1, 3, 3, 2, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 2, 3, 3, 3, 3, 2, 3 and 3 for respectively a minor, minor, mid-range, minor, minor, and major maintenance.

3. Deb's constraint handling method:

This proposes to use a tournament selection operator, where two solutions are compared at a time, and the following criteria are always enforced [23]:

1. Any feasible solution is preferred to any infeasible solution.

2. Among two feasible solutions, the one having better objective function value is preferred.

3. Among two infeasible solutions, the one having smaller constraint violation is preferred.

Table 18 shows the MSU results of five DE runs using Deb's constraint handling method

Table 18: optimal results of 5 DE runs with Deb's constraint handling method

Run	MSU
1st	0.6726
2nd	0.6648
3rd	0.6593
4th	0.6726
5th	0.6593

From table 18, the first run yields the best MSU result. Table 19 shows the optimal conditions for this method.

Table 19: Best result of DE with Deb's constraint handling method

Input: start time of ship major maintenances				Opt MSU value	Number of weeks in violation	Average Makespan	Total cost	Average availability
Ship 1	Ship 2	Ship 3	Ship 4		(>50% ship maintain/week)	(days)	(million USD)	
119	7	103	69	0.6726	0	130	7.78	73. 8461
Mode assignment with same optimal MSU value								
3 3 3 3 3 2 2 3 3 3 3 3 3 3 3 3 2 3 3 3 3 2 3 3								

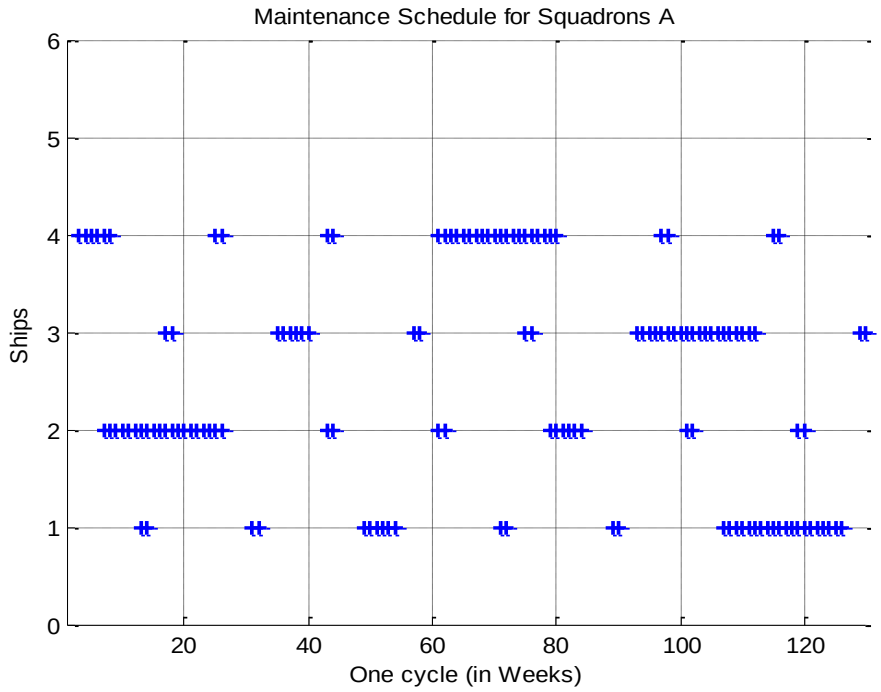

Figure 35. Graph correspond with the mode assignment in table 19

Base on the results above, we can conclude that the Deb's constraint handling method is more effective and more stable than the other methods, e.g. death penalty and penalty function methods.

4.3.3.2 PDE for Multi objective optimization

Although the weighted sum method applied in DE is very simple and easy to use, there are some disadvantages that led us to consider other methods. The first disadvantage is the difficulties in setting the weighted vectors to obtain a Pareto-optimal solution in a desired region in the objective space. Different weighted vectors need not necessarily lead to different Pareto-optimal solutions. If a single –objective optimization algorithm cannot find all optimal solution for a weighted vector, some Pareto-optimal solutions cannot be found. This part, PDE will be applied to solve the MRCSSP. PDE use the concept of dominance in its search. Unlike the

weighted sum method where the optimal solution is unique, the solutions of MOPS by PDE is a set solution for which any improvement in one objective results in the worsening of at least one other objective.

Objective functions:

Objective 1: Maximize the average availability of n ships. This is equivalent to minimizing the negative of the average availability of the n ships.

$$OBJ1 = -\frac{\sum_{i=1}^{n} avaible\ time\ of\ ship\ i}{n * total\ cycle\ time}$$

Objective 2: Minimize total cost of the n ships.

$$OBJ2 = \left(\sum_{i=1}^{n} (cost\ for\ ship\ i)\right)$$

Objective 3: Minimize the average makspan of the n ships.

$$OBJ3 = \frac{\sum_{i=1}^{n} makespan\ of\ ship\ i}{q}$$

Constraints:

$$\sum_{i=1}^{n} S_{1i,t} \leq 35$$

$$\sum_{i=1}^{n} S_{2i,t} \leq 30$$

$$\sum_{i=1}^{n} S_{3i,t} \leq 20$$

$$\sum_{i=1}^{4} P_{ik} \leq 0.5 * q$$

Where:

$S_{1i,t}$ = consumption of resource 1 by ship i in week t

$S_{2i,t}$ = consumption of resource 2 by ship i in week t

$S_{3i,t}$ = consumption of resource 3 by ship i in week t

q is the total number of ships

t is week and t run from 1st to 146th week

Parero – Optimality:

1. Definitions

- The search space or design space is the set of all possible combinations of the design variables.

- The Pareto optimal solutions achieve a tradeoff between objectives. They are solutions for which any improvement in one objective result in the worsening of at least one other objective.

- Pareto frontier: a plot of the entire Pareto set in the design objective space (with design objective plotted along each axis) gives a Pareto frontier.

- A dominated design point, is the one for which there exists at least one feasible design point that is better than it in all design objectives.

- Non-dominated point is the one where there does not exist any other feasible design point better than it. Pareto optimal points are non-dominated and hence are also known as non-dominated points.

2. Procedure for finding a non- dominated set, i.e. Pareto optimal points in this thesis.

The flow diagram shown below illustrates the procedure for finding the Pareto optimal design point for a given multi-objective scheduling problem.

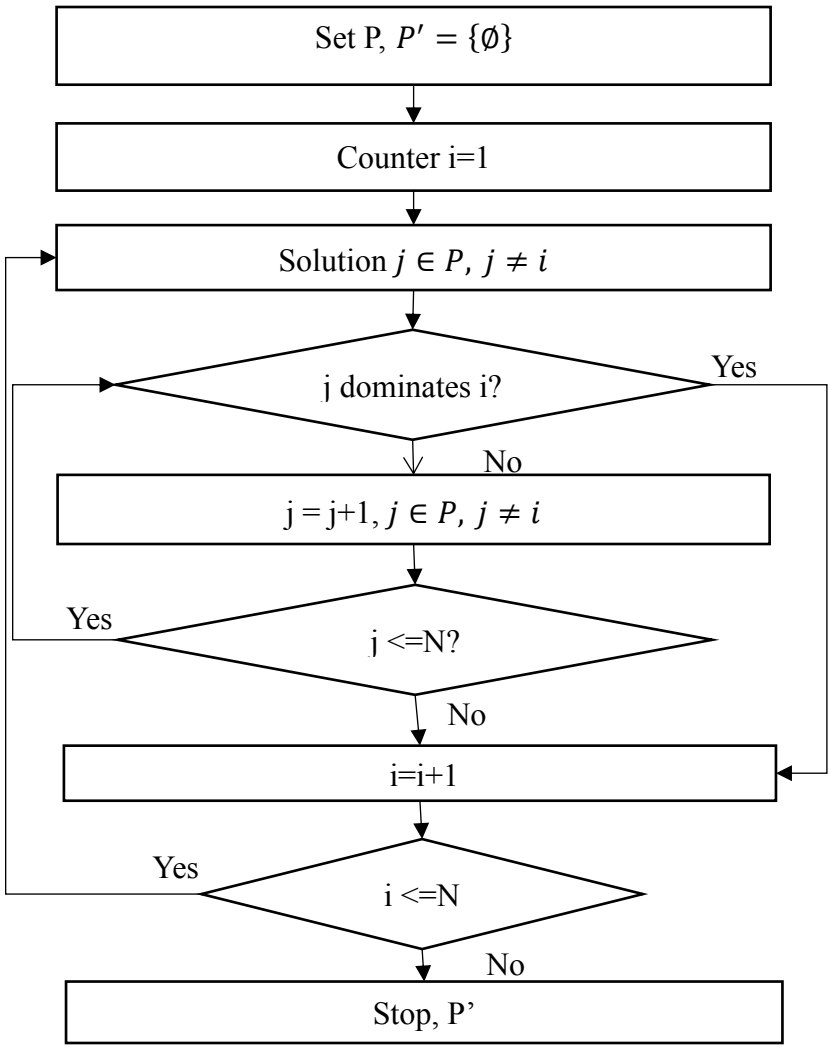

Figure 36. Non-dominated set or Pareto-optimal set P' flow chart

The notations used in this flow diagram are:

P'=set of non-dominated points

P= set of all design points.

i= a solution counter, which is essentially a distinct solution.

j= a solution counter different from i

N= total number of solutions.

Solution j dominates solution I if three parameters, i.e. total cost, average makespan, and availability of solution j are superior to the corresponding parameters of solution i, or solution j is no worse than solution I in all objectives and solution j is strictly better than solution I in at least one objective.

To illustrate the above procedure, consider the following example. Table 20 lists 11 solutions for the problem previously discussed in section 4.3.1

Table 20: Solution set for Pareto frontier determination

Solution	Total cost (USD)	Average makespan (weeks)	Negative average percentage availability (%)
1	7029500	136.25	-0.704587156
2	7172000	136	-0.705882353
3	7100500	135.25	-0.709796673
4	7273500	135.25	-0.709796673
5	7264500	136.25	-0.704587156
6	7119000	136	-0.705882353
7	7144500	136.75	-0.702010969
8	7304000	134.5	-0.713754647
9	7386500	134.25	-0.715083799
10	7258000	134.5	-0.713754647
11	7290000	134	-0.71641791

The recurring procedure for each initial solution as indicated in the major loop of the flow diagram in figure 36 works as follows:

Step 1: Begin with first solution, i.e. i=1 and P'=0

Step 2: Compare all remaining solutions for domination, i.e. j=2 to 11 with i^{th} solution. If none of these solutions dominate solution i then solution i become a member of set P'. If there is a solution j that dominates solution I then skip to the end of the procedure and restart it with another initial solution and repeat steps 1 and 2 above.

For example, when solution 1 is the reference solution, then none of the other solutions, i.e. j=2, 3 …, 11, dominate solution 1. Consequently, P'= {1} meaning the solution 1 becomes a member of P', and solution 1 is a member of the Pareto optimal frontier. The result of the above procedure is shown as table 21.

Table 21: Results of Pareto Optimal Point Procedure

Iteration	Reference solution i	None of the other solution dominate reference solution	P'
1	1	Yes	{1}
2	2	No	{1}
3	3	Yes	{1, 3}
4	4	No	{1, 3}
5	5	No	{1, 3}
6	6	No	{1, 3}
7	7	No	{1, 3}
8	8	No	{1, 3}
9	9	No	{1, 3}
10	10	Yes	{1, 3, 10}
11	11	Yes	{1, 3, 10, 11}

From the above procedure, the 4 design points that belong to the Pareto frontier as solutions.

Further experiments were run to investigate the impact of, first the population size, second, the scaling factor F, third, the cross-over probability Cr, and fourth the stability of the above procedure. The results of these experiments are shown respectively in Figures 37 – 40.

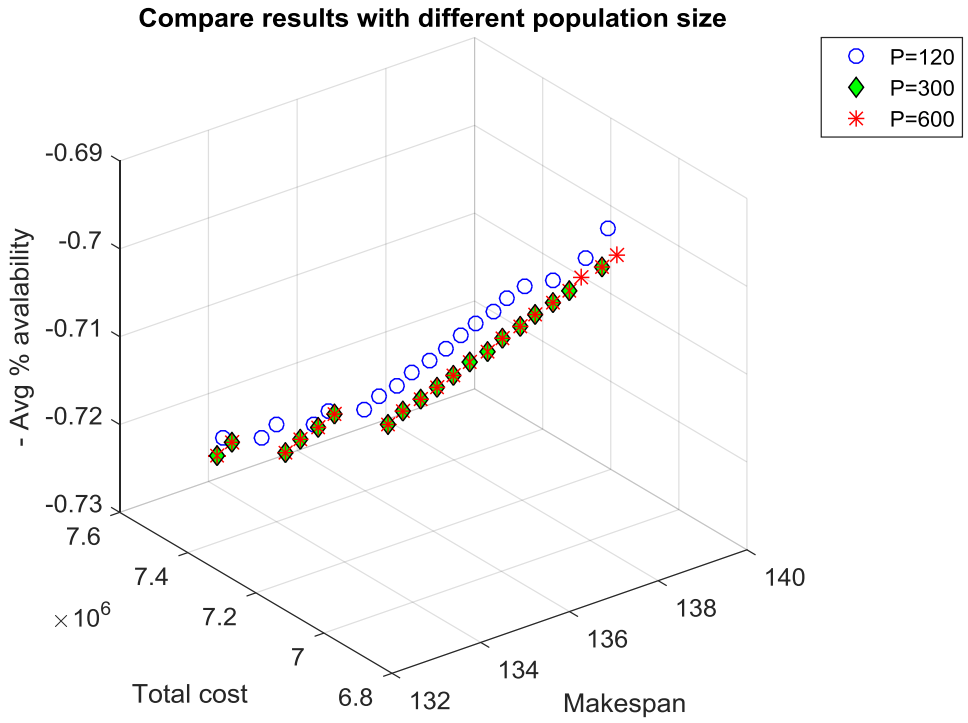

Figure 37. Impact of different population size

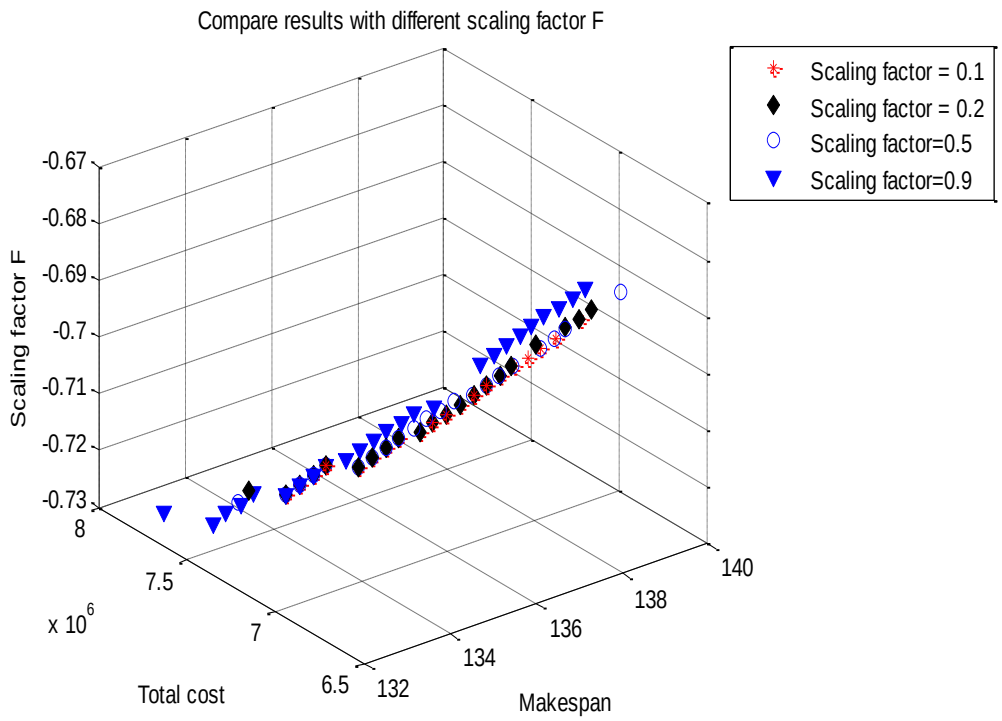

Figure 38. Impact of different scaling factor for constant crossover probability Cr=0.9

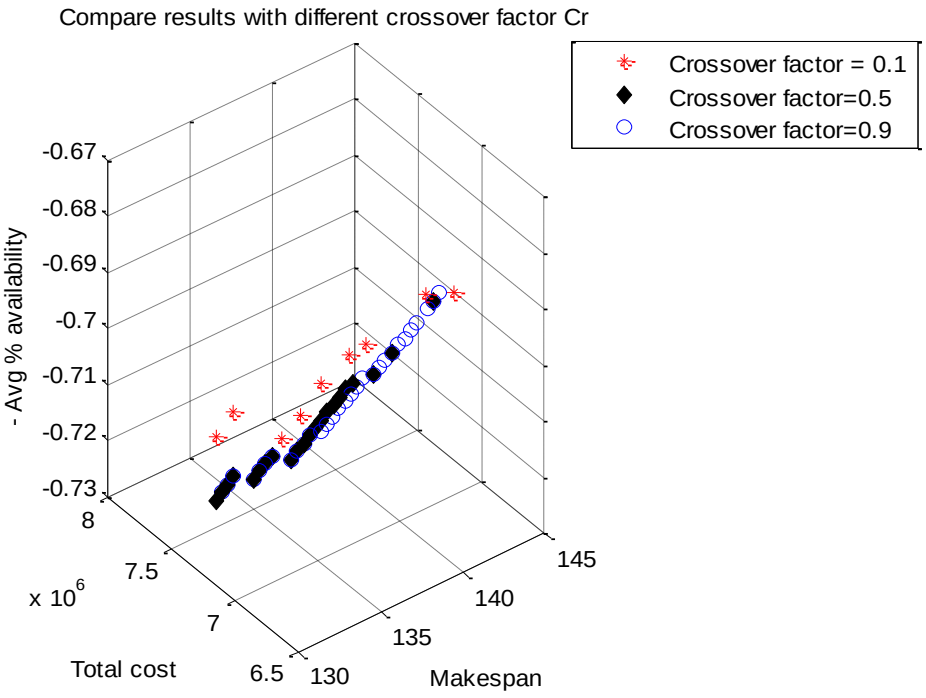

Figure 39. Impact of different crossover probability for constant scaling factor, F=0.2

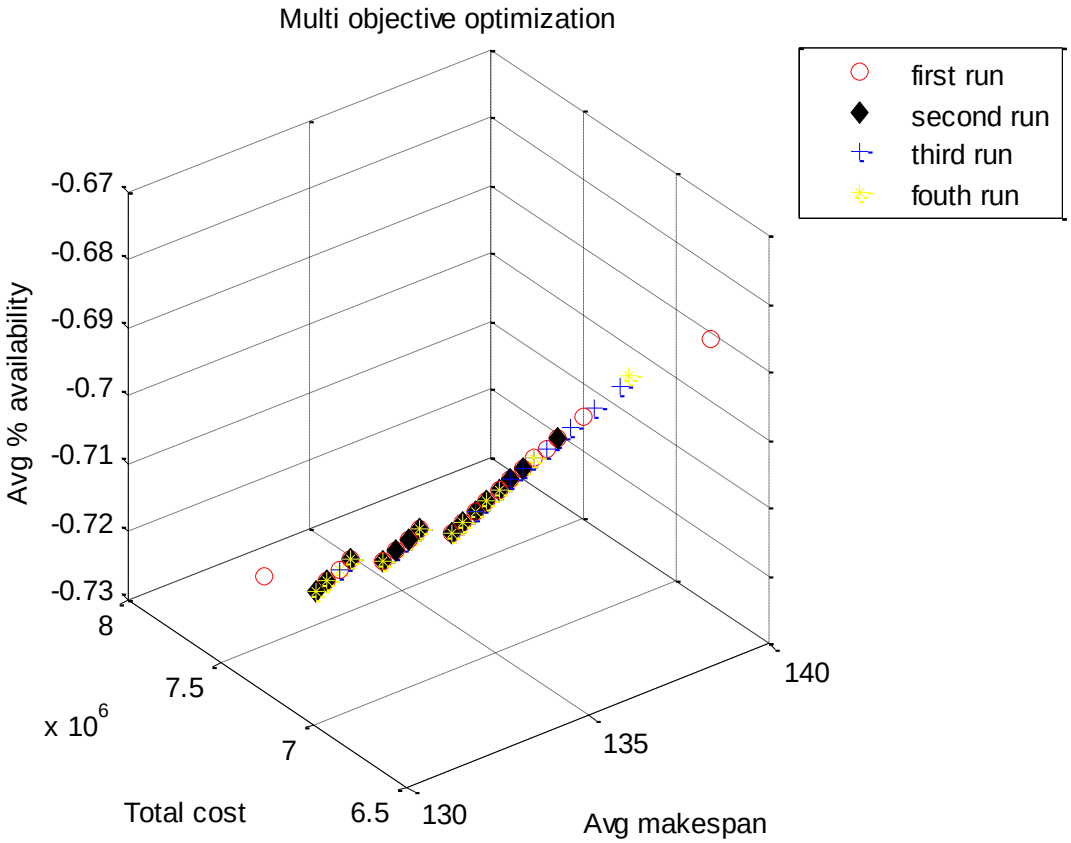

Figure 40. Stability of Pareto frontier procedure with population size 120, Cr=0.9, F=0.2

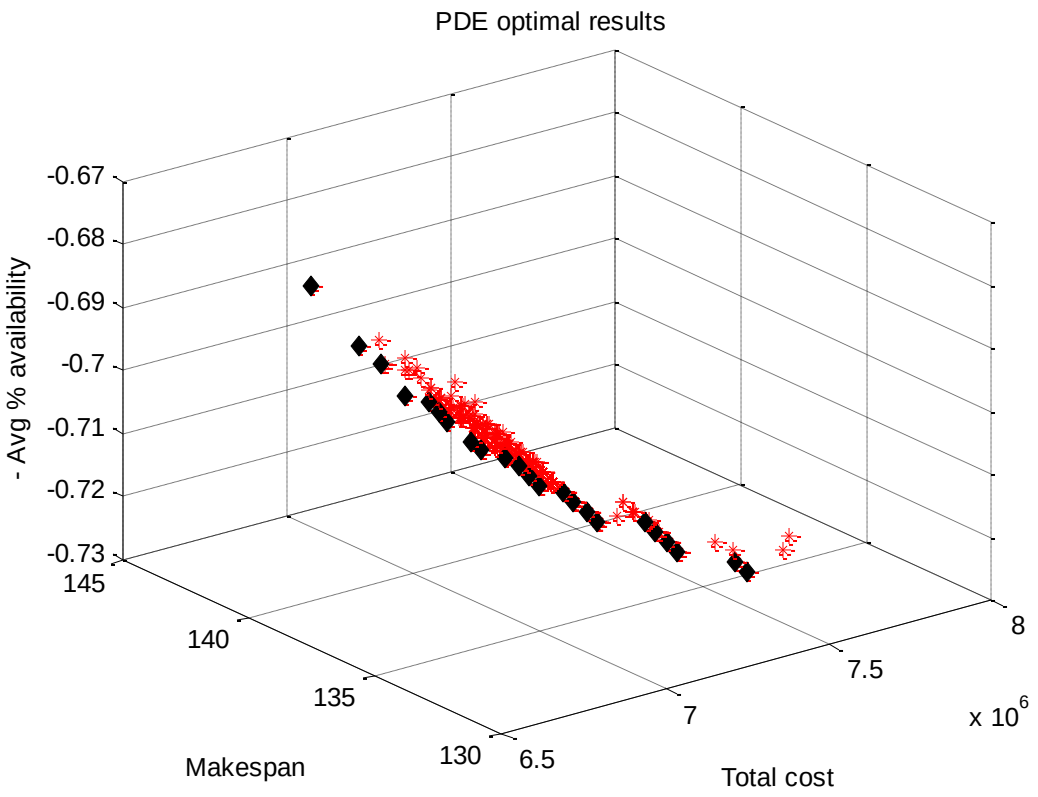

Figure 41. Include all 300 design points with those belonging to the Pareto frontier indicated by the black diamond symbols

Table 22 shows a list of the 23 optimal design points.

Table 22: List of the 23 optimal design points

Assigned Mode																								Total cost	Make span	Avg % availability
1	3	1	3	3	2	1	3	3	3	3	3	3	3	3	3	2	3	3	3	1	1	3	3	7276000	134	71.642%
3	3	1	1	3	1	1	3	3	3	1	3	3	3	3	3	2	3	3	3	3	1	3	3	7106000	135	71.111%
1	3	1	1	3	2	1	3	3	3	3	2	3	3	3	3	2	3	3	3	3	1	3	3	7263500	134.25	71.508%
3	3	1	1	3	1	1	3	3	3	3	3	3	3	3	3	2	3	3	3	3	1	3	3	7138000	134.5	71.375%
3	1	1	3	3	1	1	3	3	3	3	3	3	3	3	3	2	3	3	3	2	1	3	3	7125500	134.75	71.243%
1	3	1	2	3	2	1	3	3	3	3	3	3	3	3	3	2	3	3	3	3	1	3	3	7295500	133.75	71.776%
3	3	1	1	3	1	1	3	3	3	1	3	3	3	3	3	2	2	3	3	3	1	3	3	7093500	135.25	70.980%
3	3	1	1	3	1	1	3	3	1	3	3	3	3	3	3	2	3	3	3	3	1	3	2	7025500	135.75	70.718%
3	1	1	3	3	2	1	3	3	3	3	3	3	3	3	3	2	3	3	3	3	1	3	3	7308000	133.5	71.910%
3	1	1	3	3	2	1	3	3	3	3	2	3	3	3	3	2	3	3	3	3	2	3	3	7465500	132.75	72.316%
3	1	1	2	1	1	1	3	3	1	3	3	3	3	3	3	2	3	3	3	3	1	3	3	6993500	136.25	70.459%
3	1	1	3	3	1	1	3	3	1	3	2	3	3	3	3	2	3	3	3	3	1	3	2	7013000	136	70.588%
2	3	1	2	2	1	1	3	3	1	2	3	3	3	3	3	2	1	1	3	3	1	2	3	6875500	137.75	69.691%
1	3	1	3	3	2	1	3	3	3	3	3	3	3	3	3	2	1	3	3	3	2	3	3	7446000	133	72.180%
2	3	1	1	3	1	1	3	3	1	3	3	3	3	3	3	2	3	1	3	3	1	3	3	6925500	136.75	70.201%
1	1	1	1	3	1	1	3	3	1	2	3	3	3	2	2	2	3	1	3	3	1	1	3	6804500	138.75	69.189%
3	1	1	1	3	1	1	3	3	1	3	3	3	3	3	3	2	2	1	3	3	1	3	3	6893500	137.25	69.945%
1	1	1	2	1	1	1	3	1	2	2	2	2	3	2	1	2	2	1	3	2	1	1	1	6743500	140.75	68.206%
3	1	1	1	2	1	1	1	2	1	2	3	3	3	3	3	2	1	1	2	2	1	3	3	6779500	139.25	68.941%
1	1	1	1	3	1	1	2	2	3	3	3	3	3	3	3	1	2	3	3	2	1	1	3	6822000	138	69.565%
3	3	1	2	2	1	1	3	3	1	2	3	3	3	3	3	2	1	1	3	3	1	2	3	6888000	137.5	69.818%
3	3	1	1	3	1	1	3	3	1	3	3	3	3	3	3	2	3	1	3	3	1	3	3	6938000	136.5	70.330%
3	3	1	1	3	1	1	3	3	1	3	3	3	3	3	3	2	3	3	3	3	1	3	3	7038000	135.5	70.849%

4.4 Conclusion and discussion

In this chapter, some concepts of ship operation and maintenance activities are introduced, and their problem descriptions and model formulations are provided for optimizing the start times of each maintenance activity. A DE algorithm was developed to maximize the ship availability for two case studies, one involving single-objective optimization and the other involving multi-objective optimization. Besides ship availability, the makespan and total cost of maintenance activities are very important. Therefore, the chapter introduced a new problem for maintenance scheduling with mode assignment and resources constraints. The goal of this problem is to minimize the makespan and total maintenance cost. Over conventional mathematical methods for numerical simulation the results in Matlab proved superior to those of the DE algorithms in solving the complex scheduling problem.

CHAPTER 5

SUMMARY AND DIRECTION FOR FUTURE RESEARCH

Task sequencing and scheduling in manufacturing and service industries is one of the most critical activities to preserve the well-being and even survival of the organization in question. An effective scheme for task sequencing and scheduling means project time and cost under control and promises to the customer kept. This dissertation is an attempt to demonstrate the viability of the Genetic Algorithm (GA) and its related solution known as Differential Evolution (DE) to providing the best practical solutions to some of the most intractable scheduling problems in the industrial world. Both GA and DE are optimization algorithms inspired by natural phenomena and generally belong to the general class of direct search methods. However, they do not require the differentiability of problem functions and rely solely on stochastic idea and random numbers of search for the optimum design point or set of design points known as the Pareto frontier.

In this dissertation, there are two major studies being undertaken: mixed shop scheduling, and ship maintenance under constrained conditions.

5.1 Mixed shop scheduling

In this research, we studied the scheduling environment of job-shop (jobs with fixed technological routes) and open shop (jobs have an arbitrary machine order). In real-life, shop scheduling problems are more complicated than "pure" shop problems, i.e. job-shop, open shop and flow shop. A practical problem may be a mixture of different pure problems and is usually much more difficult. The research proposed a modified GA method to deal easily with this complex shop scheduling problem. With a new way of creating chromosomes and the method to

calculate the objective function, the proposed algorithm can handle the constraints effectively. Solutions are always in feasible regions. In addition, the proposed method is also very flexible when it can apply to both pure shop problems, such as pure open-shop and pure job-shop, and combined shop problems. The effectiveness of the proposed algorithm is tested in various benchmark problems, both in the categories of pure problems and mixed problems. The computation results showed that the proposed method obtains the optimal solutions for small size problems and demonstrates faster computed processing time, more robust than the compared methods. The near optimal solutions for bigger size benchmark problems proved that it is also practicable for a large number of jobs and machines problems. Therefore, it can be a viable alternative method to solve shop scheduling problems.

5.2 Ship maintenance scheduling

The second major part of this thesis focuses on ship maintenance scheduling problems with a Differential Evolution algorithm application. In the first two case studies, both single objective and multi-objectives optimization for scheduling problems in the presence of constraint requirements are analyzed with the DE algorithm. The goal is to find the optimal start time of the first maintenance cycle for each ship so that the number of weeks over the scheduling horizon, with fleet availability of at least 75% is maximized. In the first case study, the unique global optimal solution was found by the DE optimizer, and a set of suboptimal results were provided. In the second case study, when the ship maintenance scheduling optimization problem is formulated as a multi-objective problem, the DE algorithm can successfully solve it. The DE optimizer for multi objectives optimization problems allows the decision maker to consider tradeoffs between the conflicting objectives, then decide on the solution that best matches their preferences. The computation time of both case studies is very reasonable.

The second part also extended the ship maintenance scheduling problems to multi-mode resource-constrained ship scheduling problems (MRCSSP) by considering different execution modes of ship maintenance activities under the constraints of resources and ship availability per week requirements. The goal of the problem is to select the execution modes of ship maintenance activities to minimize the total duration or makespan of a maintenance schedule and total maintenance cost. A mix integer linear mathematic model for MRCSSP was developed but was not carried out to its end results due to the complexity of the model. The multi-objective optimization for MRCSSP were solved with two different methods in combination with the DE algorithm. The first method, the weighted sum method, transforms multiple objectives into an aggregated objective function by introducing the summation of maintenance scheduling utilities, MSUs as the objective function for MRCSSP, then applied DE to solve the problem. This method relies on the user supplied weights for each objective function to define their important level. The second method is the Pareto Differential Evolution (PDE) which applied Pareto selection to create the new population for DE and use the concept of dominance and non-dominance in its search and to define the Pareto optimal solution set. To obtain suitable parameter values (i.e. mutation and crossover probability) for the proposed algorithm, a testing process was implemented with the Golden key method. DE was also tested with different size initial populations and with three different methods for the DE to handle constraints, notably death penalty, penalty function and Deb's constraint handling method to solve the constraint global optimization problem. The results from computational experiments prove the applicability of the proposed method to the scheduling of a Navy maintenance activities. They also show that the method can be used to schedule effectively for larger projects.

5.3 Direction for Future Research

Although the research shows the advantages of evolution algorithms, i.e. GA and DE for scheduling problems, its results point to the following research possibilities to increase its robustness and to expand their capabilities:

For the mixed-shop scheduling problem: The thesis considers the mixed shop scheduling which is a combination of pure shop scheduling types, i.e. open shop and job shop. To increase the complexity of the mixed shop scheduling problem, there are several types of shop scheduling problems that can be combined, such as flow shop, dynamic shop, and flexible shop scheduling. The inclusion of the scheduling of personnel and the limited time availability for particular machines are also areas for future research.

For the maintenance scheduling problem: This research only considers one type of resource which is renewable resources, i.e. dockyard and human resources. Future research can include other types of resources, including non-renewable resource and/or partially renewable resource. In this thesis, a maintenance activity, once started, must be implemented to completion. Considering different types of maintenance activities that can be split depending on some types of cost penalties may be a subject for future research.

For GA and DE: Testing and analyzing different methods for choosing suitable parameters is necessary. Constraint handling methods is needed to increase the reliability and robustness for the algorithms. Developing the techniques for the traditional method for handling big size problems with the proposed math formulation model may be investigated. Comparing the results with a more traditional method is a more effective way to evaluate the performance of the proposed heuristics.

REFERENCES

[1] [Online database], URL: *http://www.ics-shipping.org/shipping-facts/shipping-and-world-trade*

[2] Ship, [online database], URL: *http://en.wikipedia.org/wiki/Ship*

[3] Guignier, F. and S. Madanat (1999). Optimization of Infrastructure Systems Maintenance and Improvement Policies,. *Journal of Infrastructure Systems* 5(4): 124-134.

[4] Dekker, R. (1996). Applications of Maintenance Optimization Models: A Review and Analysis,. *Reliability Engineering & System Safety 51(3): 229-240.*

[5] White, E. N. (1979). Maintenance Planning – Control and Documentation. London, Gower Press

[6] Mobley, R. K. (2002). An Introduction to Predictive Maintenance USA, Butterworth-Heinemann.

[7] Nowlan, F. S. and H. F. Heap (1978). Reliability-Centered Maintenance. *Proceedings Annual Reliability and Maintainability Symposium*, IEEE.

[8] Shields, S., K. J. Sparshott and E. A. Cameron (1996). Ship Maintenance: A Quantitative Approach. London, Marine Media Management Ltd.

[9] Richard Lee Storch, Colin P. Hammon, Howard McRaven Bunch, Richard C.Moore. Ship Production. The Society of Naval Architects and Marine Engineering, 2007.

[10] J.H. Holland. ECHO: Explorations of Evolution in a Miniature World. In J.D. Farmer and J. Doyne, editors, *Proceedings of the Second Conference on Artificial Life*, 1990

[11] Yousef Alhouli, Development of Ship Maintenance Performance Measurement Framework to Assess the Decision Making Process to Optimise in Ship Maintenance Planning, P.h.D thesis, The University of Manchester, 2011

[12] Deris, S., S. Omatu, H. Ohta and L. Shaharudin Kutar (1999). Ship Maintenance Scheduling By Genetic Algorithm and Constraint-Based Reasoning. *European Journal of Operational Research* 112(3): 489-502.

[13] Mackenzie, L. (2004). Inaugural Ship Superintendency Forum 2004, Lloyd's List Events.

[14] Mukerji, R., H. M. Merrill, B. W. Erickson, J. H. Parker and R. E. Friedman (1991). Power Plant Maintenance Scheduling: Optimizing Economics and Reliability. *IEEE Transactions on Power Systems* 6(2): 476 - 483.

[15] Ashayeri, J., A. Teelen and W. Selen (1996). Production and Maintenance Planning Model for the Process Industry. *International Journal of Production Research* 34(12): 3311-3326.

[16] Baliwangi, L., H. Arima, K. Ishida and K. B. Artana (2006). Optimizing Ship Machinery Maintenance Scheduling Through Risk Analysis and Life Cycle Cost Analysis, Hamburg, Germany, *American Society of Mechanical Engineers*, New York, NY 10016-5990, United States.

[17] Charles-Owaba, O. E., A. E. Oluleye, F. A. Oyawale and S. A. Oke (2008). Sensitivity Analysis of A Preventive Maintenance Scheduling Model. *International Journal of Industrial and Systems Engineering* 3(3): 298-323.

[18] A.S. Fraser. Simulation of Genetic Systems by Automatic Digital Computers I: Introduction. *Australian Journal of Biological Science*, 10:484–491, 1957.

[19] A.S. Fraser. Simulation of Genetic Systems by Automatic Digital Computers II: Effects of Linkage on Rates of Advance under Selection. *Australian Journal of Biological Science*, 10:492–499, 1957

[20] H.J. Bremermann., Optimization through Evolution and Recombination. In M.C.

Yovits, G.T. Jacobi, and G.D. Goldstine, editors, *Self-Organization Systems*, pages 93–106. Spartan Books, 1962.

[21] J.H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan, Press, Ann Arbor, 1975.

[22] A. P. Engelbrecht. Computational Intelligence: An Introduction' in Second Edition. *John Wiley & Sons, Ltd, Publisher*, 2007.

[23] K. Deb, An efficient constraint handling method for genetic algorithms, in *Compute. Methods Appl. Mech. Engrg*. 186 (2000) 311-338.

[24] Aytekin Bagis, Determination of the PID Controller Parameters by Modified Genetic Algorithm for Improved Performance, *Journal of Information science and Engineering* 23, 1469-1480 (2007)

[25] Andies P. Engelbrecht, Computational Inteligence: An Introduction, Wiley, John Wiley & Sons, Ltd, Publisher, 2007.

[26] P. Brucker, *Scheduling Algorithms*, fifth edition, Springer 2007

[27] V. A.Strusevich. Two-machine Super-Shop Scheduling Problem. *The Journal of the Operational Research Society*, vol. 42, No. 6 (1991), pp.479-492.

[28] N.V. Shakhlevich, Yu.N. Sotskov, and F.Werer. Shop scheduling problem with fixed and non-fixed machine orders of jobs. *Annals of Operations Research*, 92:281-304, 1999.

[29] N.V. Shakhlevich, Yu.N. Sotskov, and F.Werer. Complexity of mixed shop scheduling problem: a survey. *European Journal of Operational Research*, 120,343-351, 2000.

[30]R. Storn and K. Price, Differential evolution: a simple and efficient adaptive scheme for global optimization over continuous spaces, *Technical Report, International Computer Science Institute, Berkley*, 1995.

[31] K. V. Price, R. M. Storn and J. A. Lampinen, Differential evolution: a practical approach to global optimization, *Springer*, 2005.

[32] K. V. Price, An introduction to differential evolution, *in New ideas in optimization, D. Corne, M. Dorigo and F. Glover, Ed., McGraw-Hill*, 1999.

[33] R. Storn and K. Price, Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization, Kluwer Academic Publishers*, vol. 11, pp. 341-359, 1997.

[34] K. Price and R. Storn, Differential evolution: a simple evolution strategy for fast optimization, *Dr. Dobb's Journal*, vol. 264, pp. 18-24, Apr. 1997.

[35] R. Storn, On the usage of differential evolution for function optimization, *in Proc. the 1996 Biennial Conference of the North American Fuzzy Information*

[36] J. Lampinen, A constraint handling approach for the differential evolution algorithm, *in Proc. the 2002 Congress on Evolutionary Computation (CEC'02)*, vol. 2, pp. 1468-1473, 2002.

[37] R. Gamperle, S. D. Muller and P. Koumoutsakos, A parameter study for differential evolution, *Advances in Intelligent Systems, Fuzzy Systems, Evolution Computation, WSEAS Press*, pp. 293-298, 2002.

[38] Liu, J. and and J. Lampinen, A fuzzy adaptive differential evolution algorithm, *Soft Comput. A Fusion Founda. Methodol. Applicat*., 9, 6, 448–462 (2005)

[39] Efren Mezura-Montes, Jes ´ us Vel ´ azquez-Reyes and Carlos A. Coello Coello, Modified Differential Evolution for Constrained Optimization, *2006 IEEE Congress on Evolutionary Computation*, July 16-21, 2006

[40] Mallipeddi, R. and P. N. Suganthan, Empirical study on the effect of population size on differential evolution algorithm, *in Proc. IEEE Congr. Evol. Comput.*, 3663–3670, June (2008)

[41] Qin, A. K. and V. L. Huang and P. N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, *IEEE Trans. Evol. Comput*., 13, 2, 398–417 (2009)

[42] Ali W. Mohamed et al, An alternative differential evolution algorithm for global optimization. *Journal of Adavanced Research, Volume 3, Issue 2, April 2012*, page 149-165.

[43] Codreanu, I.,A parallel between differential evolution and genetic algorithms with exemplification in a microfluidics optimization problem. *Semiconductor Conference, 2005. CAS 2005 Proceedings. 2005 International* 2, (2005), pp. 421–424.

[44] Sentinella, M. R.,Comparison and integrated use of differential evolution and genetic algorithms for space trajectory optimization, *Evolutionary Computation,*

2007. CEC 2007. IEEE Congress on, (2007), pp. 973–978.

[45] Xu, Xing and Li, Yuanxiang,Comparison between Particle Swarm Optimization, Differential Evolution and Multi-Parents Crossover, *Computational Intelligence and Security, 2007 International Conference on*, (2007), pp. 124–127.

[46] J. Vesterstrom, R. Thomsen.A comparative study of differential evolution, particle swarm optimization and evolutionary algorithms on numerical benchmark problems. *Proceedings of Sixth Congress on Evolutionary Computation. IEE Press Piscataway. NJ. USA*. pp. 1980-1987., 2004

[47] Hegerty, B., Hung, C.C. and Kasprak, K. (2009). A comparative study on differential evolution and genetic algorithms for some combinatorial problems. *1st Workshop on Intelligent Methods in Search and Optimization - WIMSO 2009 (MICAI-2009)*.

[48] E. Taillard, BenchMarks For Basic Scheduling Problems, *European Journal of Operations Research*, 64, 1993, pp. 278-285

[49] J. E. Beasley 1990 OR-Library http://people.brunel.ac.uk/~mastjjb/jeb/info.html (accessed January 1, 2015)

[50] A. Klemmt et al (2009). Simulation-based optimization vs. mathematical programming: a hybrid approach for optimizing scheduling problems. *Computer Integrated Manuf*acturing 25 917–25

[51] L. H. Peng & S. Salim. A Modiffled Giffler and Thompson Genetic Algorithm on the Job Shop Scheduling Problem. *MATEMATIKA*, 2006, Volume 22, Number 2, pp. 91-107

[52] Price, K., Storn, R., and Lampinen, J. A., Differential evolution: a practical approach to global optimization. *Springer Science & Business Media*, 2006.

[53] George E. Dieter and Linda C. Schmidt, Engineering Design, Fourth Edition, *McGraw-Hill*, 2009

[54] https://en.wikipedia.org/wiki/L'H%C3%B4pital's_rule

[55] Curtis F. Gerald and Patrick O. Wheatley. Applied Numerical Analysis. Seventh Edition, *AddisonWesley*, 2004

[56] S. Q. Liu and H. L. Ong, Metaheuristics for the Mixed Shop Scheduling Problem, *Asia-Pacific Journal of Operational Research*, Vol. 21, No. 4, 2004, pp. 97-115.

[57] M. Pinedo, Scheduling theory, algorithms, and systems, second edition, *Prentice-Hall, Inc*. 2002

[58] N. Bumb, Job shop Scheduling using G.A. and the Giffler Thompson Approach, *Master thesis*, Department of Mechanical & Aerospace Engineering, December 2009. Supervisor: Han P. Bao

[59] P. Ghouddousi, et al, (2013). Multi-mode resource-constrained discrete time-cost resource optimization in project scheduling using non-dominated sorting genetic algorithm. *Automation in Construction* 30(2013), 216-227

[60] P. Brucker, et al (1999), Resource-constrained project scheduling: Notation, classification, models, and methods, *European Journal of Operations Research* 112 (1999), 3-41.

[61] J. Carlier and E. Pinson (1989), An algorithm for solving the job-shop problem. *Management Science* 35 164–76

[62] C. Zhang, P. Li, Y. Rao, and S. LiA (2005). New Hybrid GA/SA Algorithm for the Job Shop Scheduling Problem. *Evolutionary Computation in Combinatorial Optimization*, Volume 3448 of the series Lecture Notes in Computer Science pp 246-259

[63] J. E. Beasley, OR-Library http://people.brunel.ac.uk/~mastjjb/jeb/orlib/files/jobshop1.txt

[64] J. E. Beasley, OR-Library http://people.brunel.ac.uk/~mastjjb/jeb/orlib/openshopinfo.html

[65] P. Brucker, B. Jurisch, and B. Sievers .A branch and bound algorithm for the job-shop scheduling problem. *Discrete Applied Mathematics* 49 (1994) 107-127

[66] V. Nguyen, H.P.Bao, 2016. An Efficient Solution to the Mixed Shop Scheduling Problem Using a Modified Genetic Algorithm. *Journal of Procedia Computer Science* Volume 95, 2016, pages 475-482 Complex Adaptive Systems Los Angeles, CA November 2-4, 2016

[67] Nguyen, V., A. Kulkarni, H. Bao, M. and Kotinis. 2015. Development of an Optimization Algorithm based on Differential Evolution for the Navy Ship Maintenance Scheduling Problem. *In Proceedings of the 2015 Intelligent Ships Symposium*, 135. Philadelphia, PA, May 20-21, 2015: American Society of Naval Engineers.

VITA

Vi Hong Nguyen

Department of Mechanical and Aerospace Engineering

Old Dominion University

Norfolk, VA 23529

Education

- 09/2003 to 06/ 2008: B.S (2008), Automation and Control Engineering Technology, Department of Electro-Mechanics, Hanoi University of Mining and Geology - Hanoi, Vietnam.

Research Interests

- Primary Research Theme: Manufacturing System Design; Industrial Optimization
- Secondary Research theme: Computational Intelligence and Intelligent Systems.
- Third Research Theme: Transportation Optimization