MULTI-OBJECTIVE SPIRAL DYNAMIC ALGORITHMS-BASED FOR A BETTER ACCURACY AND DIVERSITY

AHMAD AZWAN BIN ABDUL RAZAK

MASTER OF SCIENCE

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : AHMAD AZWAN BIN ABDUL RAZAK

Date of Birth : MARCH 23, 1992

Title : MULTI-OBJECTIVE SPIRAL DYNAMICS ALGORITHMS-BASED FOR A BETTER ACCURACY AND DIVERSITY

Academic Session : SEM 1 2018/2019

I declare that this thesis is classified as:

☐ CONFIDENTIAL	(Contains confidential information under the Official Secret Act 1997)*
☐ RESTRICTED	(Contains restricted information as specified by the organization where research was done)*
☑ OPEN ACCESS	I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

_____	_____
(Student's Signature)	(Supervisor's Signature)
_____	_____
920323-06-5311	Dr Ahmad Nor Kasruddin bin Nasir
Date:	Date:

NOTE : * If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.

THESIS DECLARATION LETTER (OPTIONAL)

Librarian,
Perpustakaan Universiti Malaysia Pahang,
Universiti Malaysia Pahang,
Lebuhraya Tun Razak,
26300, Gambang, Kuantan.

Dear Sir,

CLASSIFICATION OF THESIS AS RESTRICTED

Please be informed that the following thesis is classified as RESTRICTED for a period of three (3) years from the date of this letter. The reasons for this classification are as listed below.

Author's Name	Ahmad Azwan bin Abdul Razak
Thesis Title	Multi-Objective Spiral Dynamics Algorithms-based for A Better Accuracy and Diversity.
Reasons	(i)
	(ii)
	(iii)

Thank you.

Yours faithfully,

_____
(Supervisor's Signature)

Date:

Stamp:

Note: This letter should be written by the supervisor, addressed to the Librarian, *Perpustakaan Universiti Malaysia Pahang* with its copy attached to the thesis.

SUPERVISOR'S DECLARATION

I hereby declare that I have checked this thesis and, in my opinion, this thesis is adequate in terms of scope and quality for the award of the degree of Master of Science.

_____

(Supervisor's Signature)

Full Name : DR. AHMAD NOR KASRUDDIN BIN NASIR

Position : PROJECT LEADER/ SENIOR LECTURER

Date :

STUDENT'S DECLARATION

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

_____

(Student's Signature)

Full Name : AHMAD AZWAN BIN ABDUL RAZAK

ID Number : MEL16009

Date : 1st JANUARY 2019

MULTI-OBJECTIVE SPIRAL DYNAMICS ALGORITHMS-BASED FOR A BETTER ACCURACY AND DIVERSITY

AHMAD AZWAN BIN ABDUL RAZAK

Thesis submitted in fulfillment of the requirements
for the award of the degree of
Master of Science

Faculty of Electrical & Electronics Engineering

UNIVERSITI MALAYSIA PAHANG

JANUARY 2019

ACKNOWLEDGEMENTS

Prima facia, I am grateful to Allah The Almighty for the good health and wellbeing that were necessary to complete this thesis.

I am also grateful and would like to express my gratitude to my supervisor Dr. Ahmad Nor Kasruddin bin Nasir for his invaluable guidance, constant support and continuous encouragement given throughout this project. I appreciate each and every help and commitment from him for this project to success. I am truly grateful for his progressive vision about my project and his tolerance of my naive mistakes. This project will not be success without his persistently and determinedly assistant.

I am also would like to take the opportunity to express my sincere thanks to Professor Ir. Dr. Kamarul Hawari bin Ghazali, Dean of the Faculty, for providing me with all the necessary facilities for the research and for the continuous encouragement.

Moreover, I would like to state my sincere thanks to all the lectures, administrative staffs and laboratory assistants of Faculty of Electrical and Electronics Engineering, Universiti Malaysia Pahang (UMP), who helped me in many ways throughout the 2 years of my studies. Here I also would like express my gratitude to my sponsor Ministry of Higher Education (MOHE) who granted support for me to do my research project.

I acknowledge my sincere indebtedness and gratitude to my parents for their full support, love and faith in my ability to attain my goals. I also like to indicate my special thanks to friends and batch mates of 2016 who involve directly and indirectly in this project.

Finally, again I would like to express my humble thanks to Allah The Almighty for keep me strength when I faced problems during my studies and projects. I also place on record, my sense of gratitude to one and all, who directly or indirectly, have lent their hand in this venture.

ABSTRAK

Algoritma pengoptimuman memainkan peranan penting dalam menyelesaikan banyak masalah kompleks dan masalah dunia sebenar. Penyelesaian yang dijana oleh algoritma mempunyai ketepatan yang tinggi dan boleh diharapkan. Tambahan pula, dengan perkembangan yang pesat dalam bidang teknologi perkomputeran, aplikasi algoritma pengoptimuman dalam menyelesaikan masalah menjadi lebih mudah dan semakin praktikal. Algoritma pengoptimuman juga dikenali sebagai algoritma metaheuristik yang berasal dari pendekatan heuristik. Ia adalah algoritma heuristik yang ditambahnilai dengan strategi yang maju diinspirasikan daripada pelbagai fenomena semulajadi yang ada di bumi. Algoritma ini juga dibahagikan jenis satu-objektif dan berbilang-objektif. Algoritma jenis satu-objektif boleh diaplikasikan untuk menyelesaikan masalah yang mempunyai hanya satu objektif, manakala jenis berbilang-objektif boleh diaplikasikan untuk menyelesaikan masalah yang mempunyai dua objektif. Bagi masalah kompleks yang terdiri dua objektif bercanggah yang tidak dapat diselesaikan dengan algoritma jenis satu-objektif boleh diselesaikan dengan algoritma berbilang-objektif. Penyelesaian yang dihasilkan oleh algoritma ini selalunya dijelmakan dalam bentuk lengkung pendepan-Pareto. Pendepan-Pareto yang dihasilkan adalah ukuran sebaikmana penyelesaian yang dihasilkan oleh algoritma. Ukuran paling utama adalah ketepatan pendepan-Pareto yang dihasilkan dengan pendepan-Pareto yang sebenar dan pengagihan penyelesaian sepanjang lengkung pendepan-Pareto. Setakat ini prestasi algoritma berbilang-objektif masih belum mencapai aras terbaik dalam kedua-dua ukuran tadi. Justeru itu, masih terdapat ruang untuk penambahbaikan dengan memanipulasikan strategi algoritma. Tesis ini menampilkan dua variasi algoritma berbilang-objektif berdasarkan Algoritma Dinamik Lingkaran (SDA) dengan aplikasinya untuk mengoptimumkan pengawal PD bagi Sistem Bandul Terbalik. Variasi yang pertama dipanggil Algorithm Dinamik Lingkaran Berbilang-objektif berstrategi Penyusun Tidak-terdominasi (MOSDA-NS). Variasi ini menggunakan strategi Penyusun Tidak-terdominasi dan Penjarak Kesesakan dengan SDA. Variasi kedua dipanggil Algoritma Berbilang-objektif Dinamik Lingkaran berkonsep-Arkib (MOSDA-A). Ia menggabungkan konsep Arkib dengan SDA. Kedua-dua algoritma yang dibangunkan telah diuji dengan satu set fungsi penanda-aras yang terdiri daripada 10 fungsi berbeza melangkaui pelbagai bentuk lanskap kesesuaian dan ciri. Kedua-dua ukuran ketepatan dan pengagihan penyelesaian pada pendepan-Pareto yang terhasil telah direkodkan. Kemudian, suatu analisa statistik telah dijalankan untuk mengukur tahap penambahbaikan berbanding Algoritma Berbilang-objektif Partikel Berkelompok (MOPSO) dan Algoritma Berbilang-objektif Penyusun Tidak-terdominasi Genetik (NSGAII). Keputusan daripada ujikaji menunjukkan bahawa MOSDA-NS mencapai ketepatan dan pengagihan penyelesaian yang terbaik berbanding dengan semua algoritma yang lain. Daripada segi menyelesaikan masalah dunia sebenar, algoritma yang dibangunkan telah diaplikasi untuk mengoptimumkan dua pengawal PD bagi Sistem Bandul Terbalik. Pengawal PD yang pertama adalah untuk menyingkirkan kesilapan yang terhasil daripada pergerakan linear bagi kereta bergerak manakala pengawal PD yang kedua menyingkirkan kesilapan darjah pusingan bagi bandul terbalik. Tindakbalas bagi kedua-dua darjah pendulum dan posisi kereta dalam domain masa telah direkodkan. Suatu analisa untuk mengukur prestasi tindakbalas kemudiannya dijalankan untuk mengetahui kesilapan keadaan-mantap, peratusan kelebihan-pecutan, masa kenaikan dan masa penyelesaian. Penemuan daripada analisa ini menunjukkan algoritma-algoritma yang baru dibangunkan ini mempunyai prestasi pengawalan yang lebih baik jika dibandingkan dengan MOPSO dan NSGAII.

ABSTRACT

Optimization algorithm plays an important role in solving many complex and real-world problems. Solution offers by the algorithm has high accuracy and reliable. Moreover, with the fast development in computing technology, application of optimization algorithm in solving problems is easier and becomes more practical. Optimization algorithm is also known as a metaheuristic algorithm which is originally come from heuristic approach. It is a heuristic algorithm that is integrated with an advance strategy inspired from many natural phenomena found on earth. The algorithms can be categorized into a single objective and a multi-objective type. Single objective type optimization algorithm can be applied to solve a problem with a single objective. On the other hand, multi-objective algorithm is applicable to solve a problem with two or more objectives. A more complex problem in which has two conflicting objectives where it is not solvable by the single objective type algorithm is the right type of problem for the multi-objective algorithm. Solution produced by the multi-objective algorithm is always presented in Pareto front curve representation. The produced Pareto front curve is a measure of how good the solution produced by the algorithm is. The main measurement criteria include the accuracy of the solution to the actual Pareto curve and the distribution of the found solution on the actual Pareto curve. Yet the performance of many multi-objective algorithms in terms of the accuracy and the distributed solution is not achieved at the highest performance level. There are still rooms for improvement the algorithm performance by manipulating the algorithm strategy. This thesis presents two variants of multi-objective type algorithms based on a Spiral Dynamic Algorithm (SDA) with application to optimize a Proportional-Derivative (PD) controller for an Inverted Pendulum System. The first variant is known as a Nondominated Sorting Multi-objective Spiral Dynamic Algorithm (MOSDA-NS). It is a strategy which combines a Nondominated Sorting and Crowding distance approaches with the SDA. The second variant is known as Archived-based Multi-objective Spiral Dynamic Algorithm (MOSDA-A). It is a strategy combining an Archived approach with the SDA. All the developed algorithms were tested with a set of benchmark functions comprising of 10 different functions covering various fitness landscapes and features. Both accuracy performance and distribution of the found solution on the obtained Pareto front are recorded. A statistical analysis is then conducted via a Wilcoxon Sign Rank test and a Friedman test. Both tests are conducted to verify the significant improvement of the solution obtained via the proposed algorithms to the Multi-objective Particle Swarm Optimization (MOPSO) and Multi-objective Nondominated Sorting Genetic Algorithm II (NSGAII). Result from the test shows that the proposed MOSDA-NS achieved the best accuracy and distribution performances compared to all other algorithms. In terms of solving a real-world problem, the proposed algorithms are applied to optimize two different Proportional-Derivative (PD) controllers for an Inverted Pendulum system attached on a moving cart. The first PD controller attenuates error for a linear movement of the moving cart. The second PD controller eliminates error for a rotating angle of the inverted pendulum. Transient responses of both pendulum angle and cart position in time-domain representation are recorded. An analysis on the transient responses is then conducted which measuring steady-state error, percentage overshoot, rise time and settling time. Finding of the analysis indicates that the proposed algorithms have resulted in a better control performance compared to the MOPSO and NSGAII.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

2LP	Bi-level programming
AM	Archiving method
AMGA	Archived-Based Micro Genetic Algorithm
APAES	Adaptive Strategy PAES
BA	Bat algorithm
BE	Bat Echolocation
BP	back-propagation algorithm
CF	Community Fitness
CS	Community Score
DE	Differential evolution
DMD	Diversity Metric Delta
DSD	directed search domain
EA	Evolutionary algorithm
EMO	Evolutionary Multi-Objective Optimization
GA	Genetic Algorithm
GD	Generational Distance
GP	Goal programming
HEM	hybridization-encouraged mechanism
HSCDA	Hybrid Self-Adaptive Algorithm
HV	Hypervolume Indicator
IP	Inverted pendulum
IWF	Inertia Weighting Factor
KKM	Kernel k-Means
LM	Lexicographic method
MCDA	Multi-objective Community Detection Algorithm
MCDM	Multi-Criteria Decision Making
MGA	Micro Genetic Algorithm
MLP	Multi-level programming
MLPSO	Multi-Leader PSO
MO	Multi-objective
MOA	multi-objective Algorithm
MOBA	Multi-Objective BA
MODE	Multi-Objective Differential Evolutionary
MOGA	Multi-objective Genetic Algorithm

MOO	Multi objective Optimization
MOP	Multi-objective Problem
MOPSO	Multi-Objective Particle Swarm Algorithm
MOS	Metric of spacing
MOWCA	Multi-objective Water Cycle Algorithm
MPB	Mathematical-Programming-Based
NBI	normal boundary intersection
NBIm	modified normal boundary intersection
NC	normal constraint
NDS	Non-dominated solution
NPF	No-Preference Methods
NPGA	Niched-Pareto Genetic Algorithm
NS	Non-dominated sorting method
NSGA	Non-dominated Sorting Genetic Algorithm
NSGAII	Fast Elitist Non-Dominated Sorting Genetic Algorithm
NSGSA-CM	Non-Dominated Sorting Gravitational Search Algorithm with Chaotic Methodology
OPSO	Orthogonal PSO
PAES	Pareto Archived Evolutionary Strategy
PD	Proportional-Differential
PESA	Pareto Envelope Based Selection Algorithm
PF	Pareto-front
PM	Priori method
PoM	Posteriori Method
POS	Pareto optimal solution
PSO	Particle Swarm Algorithm
RANSAC	Random Sampling Consensus
RC	Ratio Cut
SDA	Spiral Dynamics Algorithm
SKF	Simulated Kalman Filter
SO	Single-objective
SOA	single objective Algorithm
SOO	Single-objective Optimization
SOP	Single-objective Problem
SPEA	Strength Pareto Evolutionary Algorithm
SPEAII	strength Pareto EA II

SPO	Successive Pareto Optimization
STWA	Sliding Time Window Algorithm
TBP	Three-Term Backpropagation
UOF	Unique objective function
VEGA	Vector Evaluated Genetic Algorithm
WCA	Water Cycle Algorithm

CHAPTER 1

INTRODUCTION

1.1 Research Background

Optimization is defined as a methodology, an act or a process of producing something as almost perfect, effective as possible or functional (Kiranyaz *et.al.*, 2014). This method usually involves mathematical procedures like searching the maximum or minimum point of a function. This methodology is widely used in engineering and economics (Fonseca, 1995). To achieve the efficiency, a procedure which executes the solution iteratively is developed. This procedure is called as optimization algorithm (Yang, 2013). In this procedure, the algorithm compares various solution till an optimum or an approximate solution is found. An optimization model should be a tool to provide a fairly objectives decision. Optimization algorithm has become a part of computer-aided design activities as the development of computer technology is advancing (Wilde, 1962). Figure 1.1 shows three classes of optimization techniques. They are gradient-based technique, non-gradient-based technique and multi-disciplinary technique.

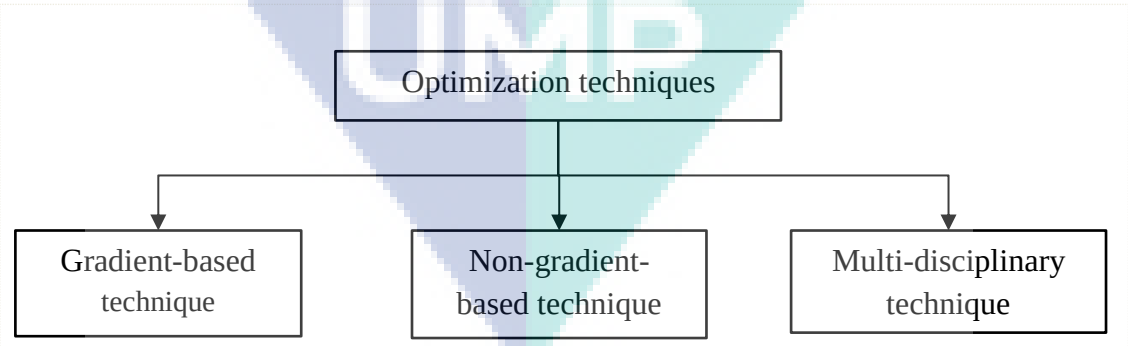

Figure 1.1 Three types of optimization technique.

Gradient based method is the one of the classical techniques which makes use of calculus and derivatives of the objective function or any constraint to search for optimum values. There are two ways to determine the optimum values either by using differential method or search method. Example of gradient-based method are including Steepest-

descent method, Newton's method, Variable-Metric method (VM method) and Conjugate-Gradient method (Yang *et.al.*, 2015). Second type of optimization technique is non-gradient-based. It also known as the direct-search technique because there is no partial derivative used. Instead, they require only function values at different points to perform the search (Stanford, 2012). On the other hand, third multi-disciplinary technique is the combination of both gradient and non-gradient-based techniques.

Heuristic and metaheuristic are types of searching strategies under non-gradient-based techniques. Heuristic is a Greek word means "discover" or "find" (Ceberio *et.al.*, 2015). In computer algorithms, heuristic defined as criteria, methodology or principles to decide which alternative among several options promises to be most effective in order to achieve an objective (Kokash, 2005). It finds the solution quicker when the classic technique is very slow, or it is applicable to find the approximated solution when the classic method failed to provide the absolute one. This is achieved by trade-off (Lakshika, 2014). The methodology make compromise of several goals, firstly is to make the objective criteria is simple and second is the tendency to discriminate correctly between minimum and maximum valuation simultaneously (Sittisak *et.al.*, 2015). To review, there are plenty of algorithms developed which belongs to heuristic algorithm including Simulated Annealing Algorithm (Algebra, 1980; Rutenbar, 1989), Tabu Search (Dorigo, 1990), Swarm Intelligent (Merkle *et.al.* 2015), Evolutionary Algorithms (Eiben *et.al.*, 2003; Sevauex *et.al.*, 2010), Neural Networks (Kriesel, 2005) and Support Vector Machine (SVM) (Evgeniou *et.al.*, 2015).

However, heuristic does not guarantee that all of the solution found are the best. Thus, the solution is only considered as approximated and not accurate solution (Kokash, 2005). This is because the algorithm needs too many information from the user and some complex integration to solve the problem. So, heuristic usually only find and provide solution close to the best agent in a fast time (Kokash, 2005). Metaheuristic on the other side is a higher-level procedure which to find, generate, develop or select a heuristic that may provide a sufficiently good solution to an optimization problem (Sörensen, *et.al.*, 2018). Their strategies are to guide the search process by exploring all the feasible region in order to find the optimal solution. The techniques of metaheuristic ranges from executing simple local search procedure to the higher level or complex procedure. This type of search are mostly approximation and commonly not deterministic (Tseng *et.al.*,

2003). Differ to heuristic, metaheuristic problem-independent techniques and have mechanism to avoid local optima (Tseng *et.al.*, 2003). Local optima are a situation where the algorithm only find the optimum value in some feasible region (Kesselring, 2006). This situation causes the possible better solution in wider feasible region could not be found. Thus, global optima are preferred. In contrast to local optima, global optima are the situation in which all feasible regions are discovered. Global optima can lead the algorithm to find the possible best solutions in all feasible region, not only in a specific region (Womersley, 2008).

The recent challenges also faced by researchers are to provide a low computation cost to reduce time of computation, as well as to reduce monetary cost to gain higher profit. At the meanwhile, the algorithm also required to provide a high accuracy solution and fast computation speed to solve real-world problem (Wang, 2016). This challenges much solved by this metaheuristic optimization method. There are two groups to classify metaheuristic algorithms, first is singled-solution-based and second is population-based method (El Yafrani *et.al.*, 2016; Lopez *et.al.*, 2018). Singled-solution-based also known as trajectory methods. The goal of this approach is to modify and improve single searching agent solution (Bandyopadhyay *et.al.*, 2013). In contrast, population-based method modifies and improves solutions from multiple numbers of searching agent. In other words, while searching the solution, they also maintain the best of found individuals in the current population.

Other than that, metaheuristic algorithms are also classified in three categories of optimization, which are single-objective (SO) optimization, multiple-objective (MO) optimization and many-objective (MaO) optimization. In SO, the finding of only one-single best solution is the goal. It determines the minimum or maximum value of a single-objective (SO) problem that combined all different objectives into a unified-function (Savic, 2002). This SO is useful to give user an insight to the nature of the problems. However, the weakness of SO is it cannot provide any other choices of solution, which are traded-off between different aims against each other. On the contrary, MO does not provide single-solution, but they provide a set of various solutions (Deb, 2004, 2014; Nain *et.al.*, 2005; Miettinen *et.al.*, 2008). This methodology, which interacting less or equal than three objective functions, can provide a set of compromised solution, whether decided by trading-off, Pareto dominance or *etc.* while MaO optimization is the

optimization which deals with more than three conflicting objectives (Palakonda *et.al.*, 2017; Bi *et.al.*, 2017; Chand *et.al.*, 2015). Some of the researcher state that, it might be consist of up to 20 objectives (Mariano-Romero *et.al.*, 2005; Chand. *et.al.*, 2015). They also found that, the number of objectives in $MaOO$ increase the challenges to balance convergence and diversity of the Pareto-front.

Since the real-world problems are complex and many aspects need to be considered, most of the designs are then characterized by a large and often infinite number of alternatives. Thus, the MO optimization is preferred by many users to identify a wide range of alternative solutions. Through this MO optimization, they do not need to predefine for which levels of the required best solutions that compared to the each other between the problem functions. Other than that, the algorithm in metaheuristic class is derivatives-free or non-gradient based (Rios *et.al.*, 2013). Derivative-free means that the algorithm not require any information of functional derivatives. This algorithm is really useful when derivative information is unavailable, unreliable or not-practical to be provided. Instead, this type of algorithm only relies on the repeated evaluations of the objective function. It iteratively searches the solution by following the certain heuristic guideline.

There are six of method-classes in MO optimization. They are including Priori method (PM) (Hartikainen, 2016b; Krause *et.al*, 2016), Scalarization (Feldman *et.al.*, 2016; Ghaznavi, 2017; Gutiérrez *et.al.*, 2017), Posteriori method (PoM) (Williamson, 2011; Al-Matouq, 2012; Bakhsh *et.al.*, 2014; Hartikainen, 2016a), Interactive method (Deb *et. al*., 2007; Miettinen *et.al.*, 2008; Sadrabadi, S.J., *et.al*, 2009; Ruotsalainen, 2010; Eskelinen *et.al.*, 2012; Dergisi, 2015; Greco, S., *et.al.*, 2016), Hybridization method (Adiche *et.al.*, 2010; Carvalho, A. L. D., 2016, and Multi-Level Programming (Osman *et.al.*, 2017). However, only Posteriori method was chosen. This method is defined as the method to find the Pareto-efficiency only which has better strategy to solve because it provide a number of traded-off solution. (Hartikainen, 2016a). A technique called as Fast Elitist Non-Dominated Sorting Algorithm (NS) is a type of concept belongs to Posteriori method. NS is the concept extracted from the extended version of multi-objective Genetic Algorithm (GA), called NSGAII (Deb *et.al.*, 2000). In particular explanation, Pareto-optimal Solution (POS) is the solution that passed through NS procedure. In this connection, POS are the solutions produced in which they are decided as Pareto-optimal,

Pareto-efficient or non-inferior, if none of the objective function can be modified for improvement in value without degrading other objective values. NS implements elitism for MO optimization, by using the elitism-preserving approach. This elitism-preserving approach is used to store all best POS found so far, beginning from the initialization of the population. The advantages of elitism is it enhances the convergence properties of the algorithm towards the Pareto-efficient set (Deb *et.al.*, 2000). Other than that, a parameter less diversity preservation mechanism is adopted in order to make sure good well-diverse of Pareto-front.

These few years back seems a lot of researchers took the opportunity to discover new nature-inspired metaheuristic algorithm (Yang, 2010, 2016; Järvilehto, 2011). This situation occurred due to the recognition of this methodology in versatility and concept. A new algorithm, named Spiral Dynamics Algorithm (SDA) is one of the recent natured-inspired metaheuristic algorithms which was introduced in 2010 (Tamura *et.al.*, 2011a). Spiral phenomena are the inspiration, which frequently occur in nature, like in nautilus shells, whirling currents and the arms of galaxies (Tamura *et.al.*, 2011b). The author is motivated by an insight that the dynamics generating logarithmic spirals might have an affinity with the effective strategy of "*diversification in the first half and intensification in the second half"* as illustrated in Figure 1.2 (Tamura *et.al.*, 2011b). Alongside with GA and PSO SDA is also a type of algorithm which derivative-free (Kennedy *et.al.*, 1995; Stanford, 2012).

(1) (2)

Figure 1.2 Illustration (1): The logarithm of spiral; Illustration (2) The spiral "*diversification at the first half and intensification at the second half*".

Fast-elitist Non-dominated Sorting (NS) approach from NSGAII is a strategy to develop good Pareto solution set for MO-type problem (Deb *et.al.*, 2000). Another method is by adopting the concept of Pareto-dominance to determine the movement of particles, at the same time the algorithm will maintain the POS found in a setup global repository (Oltean, *et.al.* 2005). This concept is taken from Multi-objective Particle

Swarm Optimization (MOPSO) also known as Archiving-method (AM) (Coello *et.al.*, 2002; Ibrahim *et.al.*, 2015). This method is easier to understand or develop as the PSO concept has some similarities with SDA. For validation, both new algorithms will be tested with various types of numerical benchmark functions in optimization. (Kaveh *et.al.*, 2011).

In order to evaluate new algorithms' performance, several classes of problems have been selected. The problem selected from wider family of optimization problems. Important to emphasize that, it is very preferable to use same algorithm to solve every problem to optimality with a minimal amount of computation. However, this will not possible. It is generally accepted that there is no perfection because there is no algorithm which can solve every problem better than any other, in which no-free-lunch (NFL) theorem supports this statement (Zhang, *et.al.*, 2015, Pan. A., *et.al.*, 2016, Wolpert, D. H., *et.al.*, 1997). In this thesis, 10 MO problems are selected which are taken from several researchers. They are come from low-dimensional MO problems which are derived by Schaffer, Fonseca and Poloni while high-dimensional MO problem which are Kursawe and ZDTs family (Antoniou, A., *et.al.*, 2007).

To know the algorithm performance in solving real world problem, a PD-controller of an inverted pendulum (IP) is used (Shaheed *et.al*, 2006). At the end of the research, two versions of multi-objective algorithms based on SDA with good accuracy and diversity in finding Pareto-optimal front will be produced.

1.2 Problem Statement and Research Gap

The problem faced by algorithms, which are faced mostly by NSGAII and MOPSO are the convergence speed towards global-optimum. From the study and experiment, both of them require less than or nearly 1,000,000 times number of function evaluation (NFE) (Nebro *et.al.*, 2008). This huge NFE is the motivation that driving the research as well as multi-conflicting problems are hard to solve. Therefore, by using metaheuristic which require high number of NFE is not compatible approach in practice. Other than that, it is good to obtain a reasonably good approximation of Pareto frontiers in a quick time rather than obtaining high quality solution in a significant long time.

Another foundation, there are still no research that has been done to develop MO-type SDA. As SDA provide a faster alongside a good accuracy solution for single-

objective problems, it may provide a better response for MO problem. Rather than that, its convergence behaviour and simple structure can lead the SDA to be one of the promising multi-objective algorithms in the future.

1.3 Research Objectives

The main objective of this study is to propose a new multi-objective optimization based on spiral dynamic algorithm that has better convergence speed, with improved diversity and accuracy in finding Pareto-front in smaller NFE. The specific objective regarding this topic are:

1. To propose a new archived-based multi-objective spiral dynamic algorithm with improved uniform diversity that requires smaller NFE.
2. To propose a new non-dominated sorting-based multi-objective spiral dynamics algorithm with improved accuracy and diversity that requires smaller NFE.
3. To apply and improve PD controller performance on an inverted pendulum system by using developed algorithms and compare them against PD-controller which optimized by NSGAII and MOPSO.

1.4 Scope of Research

The overarching aim of the research project was to develop a new multi-objective (MO) algorithm based on Spiral Dynamic Algorithm (SDA). The specific project scope are as follows.

1. In optimization, there are two types of problem, first is continuous and second is combinatorial problems. Other than that, the objective of the optimization is classified into two groups, which are single objective and multi-objective algorithm. The scope of this research is to solve continuous problem with multi-objectives.
2. From the literatures, there are six type of multi-objective techniques, which are Priori method, Scalar method, Posteriori method, Interactive method, Hybrid method and Multi-level Programming. Posteriori method are chosen due to its

superior features of Pareto-dominance to solve MO problem. Under Posteriori method, there are four approaches, which are combination of Non-dominated sorting and Crowding Distance (NS), Archive method (AM), External memory and Mathematical novel adaptive method. In this research, two of the approach which are NS and AM are the only considered approaches to be adopted to SDA.

3. To validate the performance of the algorithms, benchmark functions is used. There are various of benchmark function in literatures found including those proposed by Schaffer, Fonseca and Fleming, Poloni and Kursawe. To challenge the real-world application, ZDT family of benchmark functions are also considered. These marked benchmark functions are all used to test the benchmark functions.

4. The algorithms can be developed in various platforms. These platforms are including in MATLAB and SIMULINK, Java and Visual basic. Due to the superior of MATLAB features, it was chosen to be used to execute the proposed algorithms.

5. To know the performance in real-world problem, the proposed algorithm will be used to optimize a real-world application. An inverted pendulum (IP) system is chosen and simulated in MATLAB SIMULINK. All algorithms involved in this research are applied to optimize the IP and comparison will be made.

1.5 Publication

Some results of this research have been submitted and published for several publication in conference proceedings or journals. List of the publications are presented as follow.

Publication as first author:

1. A. Azwan AR, A.N.K. Nasir, Sha'akmal S., M. Sawal A.R. (2017) MOSDA: A Proposal for Multi-objective Spiral Dynamics Algorithm. 4th International Conference on Electrical, Control and Computer Engineering (InECCE 2017). Langkawi, Malaysia, 16-17 October 2017, pp. 15-19.

Publication as a co-author:

2. A.N.K. Nasir, A. Azwan A.R., (2017). A Multi-objective Spiral Dynamic Algorithm and Its Application for PD Design. 8th IEEE Control and System Graduate Research Colloquium (ICSGRC 2017), Shah Alam, Malaysia, 5-6 August 2017, pp, 1-6.

1.6 Thesis Organisation and Sections

The thesis is organized in five chapters. The whole thesis is described in Chapter 1. The introduction of the thesis including the research background, problem statement, objectives and scope of research. At the meanwhile, this chapter summary describes what is the thesis about in more detail and simple.

The various of literature gains are describe comprehensively in Chapter 2. By this chapter, the concept and basic of SDA will be explained well. Rather than that, the methods in MO-type algorithm are investigated and summarized. The selected methods then will be adopted into SO-type SDA and turn it into a MO-type problem solver.

In Chapter 3, the process of conceptual method adoption and programming are shown. This chapter will brief SDA in very specific with all of the equations and procedures. It will be continued with proposed MOSDAs, in which NS and A is adopted into SDA. In this chapter also, the benchmark function will be introduced and discussed.

In Chapter 4, the experimental setup is briefed including the performance metrics of the Pareto-front. The algorithm settings are illustrated well and all of the test types which listed before will be detailly described. The result which produced in graphics and numbers also presented in this chapter. Last but not least, the case study of IP and its performance result is also presented.

The last Chapter 5 will summarize all about the thesis. The significant contribution of these new MOSDAs will be explained with detail. The chapter also suggest the future work, which could make the MOSDAs better in this section.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter is about existing methods used to optimize real-world problems of complex engineering or economic system. The discussion about modifications of single-objective (SO) algorithm to multi-objective (MO) algorithm will be stressed as the research goes on how to convert SO Spiral Dynamic Algorithm (SDA) to MO SDA. Most of the citation will focus on the modification done to the GA into multi-evolutionary optimization algorithm of NSGAII, the application of Fast-elitist Non-dominated Sorting (NS) in other MO algorithms, the application and the use of various algorithms to solve MO problem. The review will be furthered to the PSO and its extension of MOPSO, the application of Archiving-method (AM) and several modifications for its improvement. Some revision on the related topic also done in order to gain more understanding in fundamentals of algorithm, deepening mathematical concept and coding language. The findings of this review will be analysed and used to generate idea on how to convert SDA into an MO algorithm using NS and AM from MOPSO. The findings then used to obtain a sequence of procedures, which can be used to develop new algorithms which require smaller number of function evaluation (NFE).

2.2 Single-objective Optimization Techniques

Single objective optimization finds only a "*best*" solution, whether minimum or maximum value of a single objective problem. The problem can be single function or a function that lumps all different objectives into only a function. The advantage of this optimisation gives good insights into the nature of the problem. However, it cannot provide a set of solution that traded-off the objectives against each other.

2.3 Multi-objective Optimization Techniques

In general, MO optimization is an area of multi-criteria decision making (Kuo *et. al.*, 2015; Almeida *et al.*, 2016; Shieh *et. al*, 2017). MO algorithms solve MO problems which involve more than one or maximum of two objective functions. These functions required to be optimized simultaneously. Hence, the optimal decision is required in the presence of trade-offs between these multiple-conflicting problems, also called as Pareto-optimal solution (POS) in decision space or Pareto-front (Henig *et. al*, 1997; Eskelinen *et. al*, 2012). This trade-off defined as a condition, where a decision done to any subject that involves losing an ability, quality, quantity or property of the subject in order to increase other criteria (Shukla *et. al*, 2012; Wu, 2017). Plus, the solution set in the Pareto-front should be evenly distributed, by achieving two goals simultaneously which is convergence to true-Pareto-front and maximizing the distribution of diversity along the Pareto-front. Other than that, MO optimization with more than three conflicting objectives is called as many-objective optimization ($MaOO$) (Palakonda and Mallipeddi, 2017).

Definition of MO problem for minimization is in Equation 2.1 to 2.2.

$$\min F(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), f_3(\vec{x}) \dots, f_m(\vec{x})) \quad 2.1$$

$$G(\vec{x}) \leq 0, \qquad H(\vec{x}) = 0, \qquad \vec{x} \in \Omega \quad 2.2$$

Denote that $\vec{x} = (x_1, x_2, x_3, \dots, x_D)$, Ω is the variable space, R^m is the objective space and $F: \Omega \rightarrow R^m$ consist of m real-valued objective functions with constraints $G(\vec{x}) \leq 0$, and $H(\vec{x}) = 0$, the feasible region is $\Omega = \prod_{i=1}^{D}[L_i, U_i]$, where L_i and U_i are respectively lower and upper boundary of the x_i.

To categorize the methods to develop MO algorithm from SO algorithm, there are six methodologies as shown in Figure 2.1. They are including Priori method (PM) (Hartikainen, 2016b; Krause *et. al*, 2016), Scalarization (Feldman *et.al*, 2016; Ghaznavi, 2017; Gutiérrez *et.al*, 2017), Posteriori method (PoM) (Williamson, 2011; Al-Matouq, 2012; Bakhsh *et.al*, 2014; Hartikainen, 2016a), Interactive method (Deb *et.al.*, 2007; Miettinen *et.al*, 2008; Sadrabadi, S.J., *et.al*, 2009; Ruotsalainen, 2010; Eskelinen *et.al.*, 2012; Dergisi, 2015; Miettinen *et.al.*, 2016), Hybridization method (Adiche *et.al*, 2010;

Carvalho, A. L. D., 2016; Ota *et.al*, 2014; Sutradhar *et.al.*, 2016), and Multi-Level Programming (Osman *et.al.*, 2017). These six methodologies will be explained later in the next section.

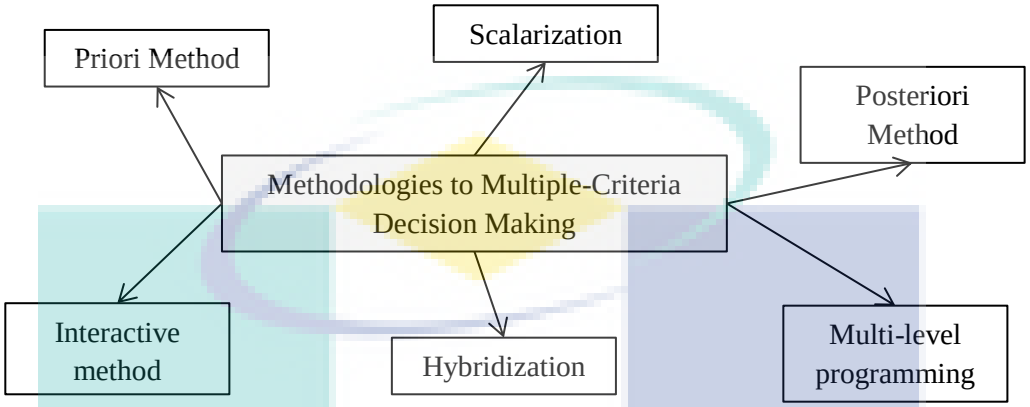

Figure 2.1 Multiple-criteria decision making methodologies.

MO algorithms are also classified into two types, which are in Mathematical-Programming-Based (MPB) and Evolutionary-Concept-Based (EA) (Eiben *et.al.*, 2003; Parrill, 2000; Szopa *et.al.*, 2011; Luenberger, *et.al.*, 1984; Eiben *et.al.*, 2015; Wong, 2015). MPB also known as classical method algorithm which run the procedure repeatedly and produces only a POS. At the meanwhile, EA mimic natural evolution and evolve a population of a solution simultaneously into a representative set of POS. Most of MPB algorithm works only to some linear or convex MO problem and some of them need make further assumption. For instance, the EA has the advantages in which they have the ability to provide the sets of solutions. This allows them to compute the approximation of the entire Pareto-front. However, the speed is the limitation for EAs compared to MPB and the POS generated are not guaranteed satisfy the Pareto-efficiency. The only known is, the POS generated are non-dominated solution.

Example of the most popular MPB include the Normal Boundary Intersection (NBI) (Lim *et.al.*, 2001; Siddiqui *et.al.*, 2012; Charwand *et.al.*, 2015), Modified Normal Boundary Intersection (NBIm) (Siddiqui *et.al.*, 2012) , Normal Constraint (NC) (Nakano *et.al.*, 2006; Piche *et.al.*, 2009; Al-Matouq, 2012; Umadevi *et.al.*, 2014), Successive Pareto Optimization (SPO) (Zhou *et.al.*, 2011; Lopez *et.al.*, 2013; Adelmann *et.al.*, 2015) and Directed Search Domain (DSD) (Erfani *et.al.*, 2013; Wang *et.al.*, 2017a, 2017b). These algorithms solve the MO problems by generating the POS through the scalarization. This will yield POS whether locally or globally. The scalarization of these

algorithms is to gain the solution of Pareto-front with evenly distributed. Example of the wide used EA are Fast-Elitist Non-Dominated Sorting Genetic Algorithm (NSGAII), strength Pareto EA II (SPEAII) (Zitzler, *et.al.*, 2001; Deb, 2014; 2001; Zitzler *et.al.*, 2004; Deb *et.al.*, 2005; Shukla *et.al.*, 2012), MOPSO (Parsopoulos *et.al.*, 2008; Coello *et.al.*, 2006; Abido, 2009; Liu *et.al.*, 2013; V. Kumar, 2014; Jia *et.al.*, 2017) and Stimulated Annealing (Suppapitnarm *et.al.*, 2000; Bandyopadhyay *et.al.*, 2007; Li *et.al.*, 2011; Naderi *et.al.*, 2011). Denote that NSGAII and SPEAII have become the standard approaches widely used in many applications or optimization software. Recently, a new method based on EA was introduced. The new novelty-based paradigm for MO problems is the improvement for algorithm in guiding the exploring agents to the unexplored region (Kuhn, 1970; Corucci *et.al.*, 2015). This is to reduce the bias and plateaus as well as the guide the searching phases in MO algorithm. At the end of the iterations, PoM will provide a finite POS that "dense enough" or a set of POS which may contain the approximated vectors of the optimal solution.

2.3.1 Priori Method

Before the solution is processed, this method needs all of the sufficient information. This type of procedure is called as Priori method (PM). First example of PM is Lexicographic method (LM) (L. Jaimes *et.al.*, 2011). This LM define all the objectives of the MO problems can be arranged and ranked in order or the importance (Zhukovin *et.al.*, 1988). LM assumes that, for all objective function without loss of generality, can be ranked in the most important to less important, for example Function 1, f_1 is the most important to the k^{th} number of functions, f_k less important objective. For instance, there are many mathematical and reasoning approach for this LM including monoids of word, Cartesian product, over a well-ordered set functions, finite subsets, group of order (Z^n), colexicographical order, and monomials (Ogryczak *et.al.*, 2006; Ogryczak *et.al.*, 2007; Castro-Gutierrez *et.al.*, 2009; Marques-Silva *et.al.*, 2011; Orumie *et.al.*, 2013).

On the other hand, another PM called Goal Programming (GP) is introduced also apply same concept as in LM (Gorges *et. al.*, 2005). However, in GP, a goal or a target need to be achieved (Gass, 1987; Li, 1996; Ignizio, J. P., 1976; Gorges *et.al.*, 2005; Tang *et.al.*, 2012; Gupta *et.al.*, 2018). An achievement function will be setup in order to minimize the unwanted deviation from a set of the goal. The achievement function could be a vector or weighted-sum, which depends on the variants of the GP applied. To satisfy

all of the functions, an underlying satisficing philosophy is assumed. According to the literature, GP has been applied to determine the needed resources to obtain the desired set of goals, determine the degree of POS with the resources and providing the best POS based on the resources or the priorities itself. This GP then furthered study in 1995 to explain the detail of this state-of the art MO algorithm (Li, 1996).

As conclusion, PM is a simple method. This is the main advantages that this method has. This priori method faced no problems to solve large number of variables, objectives and constraint. However, the main critic is the ability of PM to provide solution, which is not Pareto-dominance (Ma *et al.*, 2015; Hartikainen, 2016b). This critic is against the concept of decision-making theory, which state that the no rational decision making will clearly provide the solutions, or the solutions provided by this method are concluded as not satisfying conceptual of Pareto-efficiency.

2.3.2 Scalar Method

One of the method in MO optimization is scalarization technique (Pagani *et al.*, 2009). In scalarization, the procedure is to formulate an SO problem by assuming the optimal solution provided by the SO algorithm is the Pareto-optimal solution to the MO problem (Eichfelder, 2009). Another definition, the scalarization is the method of combining all of the functions in any MO problem into scalarized one-objective function. Some of the researchers known this method as the weighted-sum because all of the functions are minimized through the new generated unique objective function (UOF). When the UOF is optimized, the image of the graph shows that the solutions are in the form of Pareto-front. At the initial state, the user needs to define the desired constant of weight vector. This is important in order to obtain a solution with strict-Pareto or weak-Pareto, depends on the application of the MO problem. This two criteria are actually representing which part of the Pareto front the solution belongs to, and which type of solution are more preferred by the user (Ghaznavi, 2017; Gutiérrez *et.al.*, 2017).

An approach, called No-Preference Methods (NPM) is also based on the scalar method. The NPM does not require any preference of the MO problem which did not explicitly articulate by the user. The most known example is the scalarized-global criterion (Feldman *et.al.*, 2016). However, this method is quite sensitive to the scaling of

the objective functions. Hence, the author suggested that MO problem are better normalized into uniform or dimensionless scale (Eskelinen *et.al.*, 2012; Ghaznavi, 2017).

2.3.3 Posteriori Method

Posteriori method (PoM) is the most used MO optimization approach categorial used to optimize MO problem nowadays (Hartikainen, 2016a). This PoM is the method that provides POS or the representative subset of the Pareto-front (Bakhsh *et. al.*, 2014). This set of POS is defined as the approach or state of allocation, which is impossible to reallocate so as to make any of criterion better off without making at least one criterion is worse-off (Aziz *et.al.*, 2015). The determination of Pareto-efficiency in engineering and economics seems particularly useful. By having all of the optimal solutions, a solution designer will focus trade-offs within the objectives, rather than they required to focus on full-ranges of the objectives function.

To explain Pareto-dominance concept in mathematical way, let be two vectors of $\vec{u} = (u_1, u_2, u_3, \dots, u_m)$ and $\vec{v} = (v_1, v_2, v_3, \dots, v_m)$.

1. $\vec{u}$ is said to be weakly dominates $\vec{v}$, if all $\vec{u}$ is less or equal to $\vec{v}$. This definition can be expressed as in Equation 2.3.

$$\vec{u} \preccurlyeq \vec{v} \text{ if } \forall i : u_i \leq v_i \tag{2.3}$$

2. $\vec{u}$ is said to be dominates $\vec{v}$, if all $\vec{u}$ is less or equal to $\vec{v}$ and there is exist of vector $\vec{u}$ which less than $\vec{v}$. This expression can be expressed as in Equation 2.4.

$$\vec{u} \prec \vec{v} \text{ if } \forall i : u_i \leq v_i \text{ and } \exists i : u_i < v_i \tag{2.4}$$

3. $\vec{u}$ is said to be strongly dominate $\vec{v}$, if all of vector $\vec{u}$ are less than $\vec{v}$. This expression can be expressed as in Equation 2.5.

$$\vec{u} \prec\prec \vec{v} \text{ if } \forall i : u_i < v_i. \tag{2.5}$$

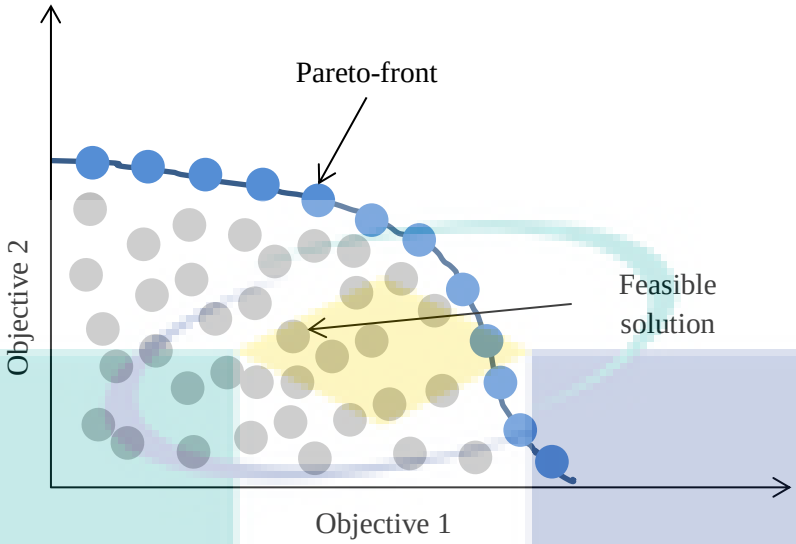

Figure 2.2 Pareto-front illustration which is optimized to maximum.

Product of Pareto-dominance criteria decision will be formed in a graph or in Pareto-front as illustrated in Figure 2.2. In this figure, the example of maximization of a MO problem is shown. Pareto-front, which consist of POS or also can be called as non-dominated solution will be formed while other solutions which do not suit Pareto-dominance criteria will be called as feasible solutions. Most of time, only POS is shown, but the feasible solutions are not shown and totally ignored.

To measure the performance of the POS, the authors provide some criteria. The first measurement is the cardinality or the number of POS found, diversity or distribution and the spacing (Laumanns, 2002; Ma *et.al.*, 2015; Tian *et.al.*, 2016). Hypervolume indicator is the one example of the method that could be applied to measure performance of Pareto-front (Bradstreet, 2011; Cao *et.al.*, 2015). This indicator will simply measure the multi-dimensional volume of the hypercubes which will be generated in the objective spaces. These hypercubes will cover all the POS in the feasible region. The authors also stated ϵ-constraints also can measure the performance through the Pareto-front. (Bérubé *et. al.*, 2009; Hu *et. al.*, 2013). The performance tests will be explained in the next section with detail.

2.3.4 Interactive Method

This method needs the user to define their initial preference for each loop of the iterations (Bandyopadhyay *et.al.*, 2008; Miettinen *et.al.*, 2008; Eskelinen *et.al.*, 2012;

Hartikainen, 2016c; Miettinen *et.al.*, 2016). While in the searching process, the user is continuously interacting with the algorithm in selecting the POS. This is not a preferable method for most of the application because this method has the weakness in which the solution might be wrongly chosen, as the user will stop the searching manually if they felt that the desired solutions were found. To decide the solution, there are three types preference of information. The solution could be based on the trade-offs information, reference points or the class of the MO problem (Ruotsalainen, 2010).

For the first preference of trade-offs, the user will be shown the current solution which traded-off among the objective functions (Garcia *et. al.*, 2011; Dergisi, 2015). The user will decide whether the solution is likeable or vice-versa. The second types, preference point is the desired value of each objective function. The user will set this value and the algorithm will continuously show the user the solution for this reference point for each loop. For the third type of preference, the user need to specify the preference of the current POS into different class indicating how the values of each objective should be changes for a better solution. This classification information are considered when the more preferred POS is computed (Deb K., *et.al.*, 2007).

2.3.5 Hybrid Method

Recently, many researchers hybridize the algorithm with various concept in order to deal with the MO problem. Hybridized MO algorithms are numerous. Hybridization in the MO optimization is the application of the combining method or algorithm between two different types, *e.g.* MPB plus EA. There are numerous of hybrid methods, however the hybridizing Multi-Criteria Decision Making (MCDM) (Mnasri *et.al.*, 2013) and Evolutionary Multi-Objective Optimization (EMO) (Lotfi *et.al.*, 2017) are the best example. However, this method has the weakness of computation cost. Most of the literature reviewed, the combination of two or more algorithms produce a complex structured MO algorithms (Goel *et.al.*, 2002; Adiche *et.al.*, 2010; Baraskar *et.al.*, 2013; Bautista *et.al.*, 2013; Carvalho, A. L. D., 2016; Ota *et.al.*, 2014; Kuroda *et.al.*, 2015; Aider, 2015; Yang *et.al.*, 2015; Borhanifar *et.al.*, 2015; Colledani *et.al.*, 2015; Englander *et.al.*, 2015; Gorjestani *et.al.*, 2015; Ismail *et.al.*, 2015; Oesterle *et.al.*, 2016a, 2016b; Krause *et.al.*, 2016; Lotfi *et.al.*, 2017; Lu *et.al.*, 2017). However, this method also has the high possibilities of providing the best accuracy and diverse Pareto-front, depends on the method or strategy involved.

2.3.6 Multi-level Programming

This method is aimed to find the only one best optimal points from the Pareto-front (Bialas *et.al.*, 1980). Multi-level programming (MLP) arranges the MO problem according to the hierarchy (Bialas *et.al.*, 1980; Gaur *et.al.*, 2008; Ansari *et.al.*, 2011; Arbaiy, N., *et.al.*, 2012). The MLP will minimize first to n^{th} objectives, to find small optimal sets for each objective. The MLP is useful when the user is not interested in the trade-offs between the functions or when the hierarchical order is the preference. However, this method could provide a largely constrained hierarchy of Pareto-front solution, which also could be infeasible. This means that the less important objective function is less considered or tend to have absolutely no influence on the last given solution. One of MLP algorithms is bi-level programming (2LP). 2LP only concerned for only two optimization problems. The first level called upper-level problem is the feasible region which determined by the knowledge gained for other optimization problem called lower-level (Ansari *et.al.*, 2011). The problem is modelled by means the first problem are constrained according to the POS from the lower-level problem. In general, 2LP is a strategy to find the POS with two decision makers, which the first level is constraints by the second level of feasible region, then the second level is constrained by the third level and so forth. 2LP is also related to the van Stackelberg equilibrium problem and the mathematical approach of equilibrium constraints. Recently foundation, better and complex 2LP have been developed as the current method is really hard to solve. This is the main weakness of the 2LP, which increases the computation cost, at the same time provides a solution which not too satisfy the Pareto-efficiency (Shih, *et.al.*, 2003; Cheng *et.al.*, 2013; Sen *et.al.*, 2013; Osman *et.al.*, 2017).

2.4 Multi-objective Algorithms

There are numerous of multi-objective algorithms that proposed by researchers for the last three decades. These algorithms are derived and introduced to handle many problems in real-world (Aronsson *et.al.*, 2006). As the world's technology grow fast and peoples competing each other, the technology needs to be always improvised in order to become better. The MO algorithms play a big role to these races as it is a part of the technology. Researchers try hard to provide algorithms that can provide a Pareto-front solution with high accuracy and diversity. Differs to SO algorithm, MO algorithm does not provide a single solution (Jensen, 2003). PSO and GA are some of the most popular

SO algorithms that used in the world today. These two algorithms are good in term of their structures, while its extended version, MOPSO and NSGAII are the widely used multi-objective algorithms to solve multi-objective problems.

As mentioned before, Pareto-dominance is the main concept that applied to solve more than one objective problems. It will provide the best solution that may be achieved by determining the outcome of Pareto-game theory, which the result will be considered as Pareto-dominated, if some of the outcomes would make at least one player better off without hurting another player (Aziz *et.al.*, 2015; Pardalos *et.al.*, 2008). The Pareto concept then applied on many computing algorithms by researchers such as ones developed in year 1994 called Niched Pareto Genetic Algorithm (NPGA) (Horn *et.al.*, 1994). NPGA is among of the early algorithms that directly addresses the diversity of the approximation set. The addition element in NPGA added to traditional GA are localized in the selection mechanism. In NPGA, to make a constant a diversity of population, the modified tournament selection called Pareto-domination tournaments along with fitness sharing in the objective space is used. However, NPGA becomes inferior compared to other newer algorithms. For example, the comparison investigated by Zitzler (2013), the NPGA ranked at number fifth out of six multi-objective Evolutionary Algorithm. The concept of Pareto-domination tournaments along with fitness sharing also used in NPGAII, NSGA and MOGA (Murata *et.al.*, 1995).

NPGAII is derived from previous NPGA with the main improvement of the usage degree of the domination. Degree of domination is the number solutions in the current population that dominate it, functioning as the determination of score in the tournament selection (Erickson *et.al.*, 2001). This method leads to noisier search. However, the NPGAII brings along with the following drawbacks, which is this algorithm was only tested to an *SO* problem and random function. It can simply be said that it has limited of proved performances compared to other algorithms. It cannot be summarized as powerful than other if it was not tested many times and included with the statistical result. Another characteristic of NPGAII, it has a high sensitivity towards the parameter controlling tournament selection and fitness sharing. Some of literatures stated, an updated fitness sharing strategy may be applied in order to avoid chaotic perturbations in the population composition (Kunkle, 2005).

In year 1994, Srinivas and Deb introduced Non-Dominated Sorting Genetic Algorithm (NSGA) (Ghiasi *et.al.*, 2011). NSGA is similar to NPGA and NPGAII in most of their sequence, but NSGA only differs in the method used to rank population in the tournament selection. The main idea of non-dominated-sorting method in ranking selection is to obtain good points while a niche method is used to maintain a stable subpopulation of these good points (Ghiasi *et.al.*, 2011). As the year changes and becomes better, the algorithm was then also surpassed by other state-of-art algorithms.

Fast-elitist Non-dominated Sorting Genetic Algorithm (NSGAII) is very important in this research. Originated from NSGA and proposed in year 2000, there are a lot of modifications done to develop modified-NSGAII for various multi-objective optimization problems, in order to create algorithm that have better performance according to the various specific applications needed whether in engineering, economics or sciences (Srinivas *et.al.*, 1994). The modifications of NSGAII were created from the original NSGA, which is more complex in three terms of the step involves in the optimization sequence. In most criteria, NSGAII have a lot of differences compared to the original NSGA, but the authors kept the term Non-Dominated Sorting (NS) to highlight its genesis and place of origin. The NSGAII solved these following weakness or drawback of NSGA which are:

1) First, high computational complexity of NS, $O(hY^3)$, M = number of objectives, N = population size. The improvement is, the NSGAII approach will require at most of $O(MN^2)$ calculation.

2) Second, lack of elitism. By improving elitism such as in NSGAII, performance of the GA can be faster. It also at the same time can avoid from the loss of good solution.

3) Third, need of specifying the sharing parameter, σ share. The need of specifying the sharing parameter in NSGA was improved into non-parameter diversity mechanism which overcome this problem. NSGAII is superior during latter generations regardless of the level of noise presence in the problem.

Because of NSGAII is the one of the superior multi-objective algorithms, there are a lot of modifications done to solve many specific applications. One of the

modifications was done by Ibrahim (2014). They hybrid the NSGAII with the back-propagation algorithm (BP). They used this hybrid-NSGAII to optimization of the accuracy and complexity of the three-term backpropagation (TBP) network. They managed to improve the classification performance with a smaller number of hidden nodes. Previously Yijie (2008), improved NSGAII based on hybridization-encouraged mechanism (HEM-based NSGAII). In their paper, they improved the convergence and diversity by using this method. HEM technique is used to normalize distance to evaluate the different genes in the population. HEM-based NSGAII used to design a longitudinal flight control system, which show that HEM-based NSGAII able to provide reasonable and correct Pareto-front. For better understanding of NSGAII, some application which used original NSGAII also reviewed. Among the application, a group of researchers applied NSGAII to optimize the placement process of switching devices in distribution network (Mazidi *et.al.*, 2013). Other than that, Shen (2008) used NSGAII to optimize urban road traffic signal and Jiang (2016) optimize a laser welding process parameters of stainless steel 316L.

NSGAII also used to estimate homograph (Osuna-Enciso *et.al.*, 2016). The author in their paper shows how NSGAII operated to estimate the homographs between two different perspectives that hold a very large set of abnormal data. The random sampling consensus (RANSAC) method used to maximise the number of matching points of given a permissible error (pe) (Fischler *et.al.*, 1981; Chum *et.al.*, 2005). The conflict was when the pe value is low, the accuracy is increased but the ability of its generalization that refers to the number of matching points that tolerate noisy data is degraded. This problem was solved successfully using NSGAII.

MOPSO is proposed at first in late 90's (Moore *et.al.*, 1999; Coello *et.al.*, 2002). This early-stage-modified version of PSO then become the motivation to other developers to extend this version of MOPSO (Coello *et.al.*, 2002). This version of MOPSO however becomes the most referred version. The technique used in MOPSO is by initializing the particles population characteristic as "Best position". This denote the best experience or the best fitness value obtained by the searching agents. These values then will be used to store the POS generated previously. Based on PAES, a global repository is setup (Knowles *et.al.*, 1999). This repository acting as the storage for POS. The particles deposit their movement experience for each iteration. The mechanism of global attraction

combines the previous found POS that lead towards globally POS. From most of the literature, MOPSO is an algorithm that able to provide a diverse and accurate Pareto-front solution (Coello *et. al.*, 2006). However, the main problem face is the archive size increases very quickly per iteration. The archive that need to be updated for each iteration leads the computing cost become higher.

There is also a modification to MOPSO done in 2015. Ibrahim (2015) proposed multi-leader PSO (MLPSO). In this MLPSO, the author developed MOPSO version in which the movement of the particles or searching agents are determined by all of the leaders that dominate the particles. MLPSO replaced the concept of uni-leader in MOPSO, by having multiple numbers of leaders. By this method, the particles will have better information gained by sharing. In this paper, the author also denoted that they did not compare MLPSO with the original MOPSO. However, they include the random leader selection to the MOPSO without include multiple-leaders concept (MOPSOrand). From the result, MLPSO outperformed MOPSOrand at all test. The level of the improvement is also significant. As the conclusion the multi-leader's concept are relatively good to have an agent that rich in information.

Multi-objective Water Cycle Algorithm (MOWCA) originated from SO Water Cycle Algorithm (WCA) (Sadollah *et.al.*, 2015). This algorithm is the one of the newest algorithms introduced. The WCA first introduced was inspired from the observation of water cycle process, flows of rivers and streams to the sea in the reality (Sadollah *et.al.*, 2016). The MO approach adopted to the WCA in order to solve MO problem in this paper is the same concept of NS which also adopted in NSGAII. The NS suit the WCA concept very well and this algorithm reveals that it can provide an impressive Pareto-front solution based on the statistical result from the stated performance metric. However, the robustness and the exploratory of MOWCA depend highly on the nature and the complexity of the problems, which is high in computation cost.

Differential evolution (DE) is the algorithm that solve SO problem introduced by Storn (1995). The concept of DE uses a simple mutation operator based on differences between pairs of vectors of solutions. These vectors are used to find the direction of searching agents based on the distribution of solution. Other than that, DE also adopted a concept called steady-state-like replacement mechanism. In this mechanism, the new generated agents or trial vector competes only against its corresponding parent or old

vector and replace their parent if they had higher fitness value. DE is same with some previous EAs, which it is a population-based approach, recombination and mutation are the operators function to generate new solution (Storn *et. al.*, 1997). The extension of DE, MODE introduced a Pareto-based approach in order to implement the selection of the best individuals (Babu and Anbarasu, 2006). The biggest difference between DE and MODE was the MODE did not compare the trial vector with the corresponding parent vector. Instead, both of the populations are combined. After the ranking of the global position is done, then the crowding distance (CD) is calculated. The advantage of MODE can be concluded as it provides Pareto-front with a great diversity. However, the run-time complexity increases as the population size larger.

From the author Ziztler, Evolutionary Algorithms (EA) are often most applicable to solve multiple-conflicting objectives (Zitzler *et.al.*, 1999). They then introduced A Strength Pareto Evolutionary Algorithm (SPEA) in year 1999. This SPEA combines several features of previous MO algorithm in very different way. SPEA method include 4 steps to search POS, 1) storing POS externally for a while, and the population will continue to be updated, 2) evaluate the fitness value of an individual, depend on the number of external POS points that dominate it, 3) preserve the diversity and 4) incorporating a clustering procedure in order to decrease number of POS without destroying its properties. The result of Pareto-front obtained from SPEA show that it capable to guide the search towards the Pareto-optimal front. However, SPEA should be improved in terms of its ability to adopt other method of solution set searching, the ways to evaluate comparison of the POS and SPEA should also be tested with more MO problems. After that, SPEA2 was then introduced in 2001 by improving method to prune the size of archive that retain the boundary solutions in that archive (Zitzler *et.al.*, 2001). From the literature, SPEA2 is found as more superior compared to NSGAII, at most in term of searching in high dimensional search spaces. SPEA2 has improved in three different aspect compared to SPEA (Zitzler *et.al.*, 1998). 1) It incorporates a fine-grained fitness assignment strategy, 2) Using a nearest neighbour density estimation techniques and 3) It has enhanced archive truncation method.

Bat algorithm (BA) is one of the latest metaheuristic algorithm which proposed (Yang *et. al.*, 2013). BA inspired from the echolocation behaviour of microbats, which emitting with various pulse rate and loudness. The extension of BA, multi-objective BA

(MOBA) is then introduced by the same author (Yang, 2012). MOBA applied the Pareto-optimality concept that used in NPGA. The empirical result show that the MOBA is also an effective solver for MO problem. It provides POS that accurate and diverse compared to true-Pareto-front. However, the additional test is highly needed. From the analysis, this original MOBA cannot handle diverse range of problem, which mean that the MOBA only limited to specific problem only.

Pareto-archived evolution strategy (PAES) is a simple EA proposed back in year 1999 (Knowles *et.al.*, 1999). In PAES, mutation operator make parent generates only one offspring. The comparison then is done to determine whether the offspring dominates the parent or not. If dominates, the offspring will become new parent and the iteration will be continued. Vice-versa, the offspring will be discarded if the parent more dominates. If both not dominates each other, previous POS will be compared and used. PAES then modified by Oltean (2005) using adaptive strategy to form Adaptive PAES (APAES) in order to ensure a better exploration. From the experiment done by them, the APAES performed as well as compared to the original PAES. However, APAES still had a slow computation time rate is similar with PAES.

Non-dominated Sorting Gravitational Search Algorithm with Chaotic Mutation (NSGSA-CM) was proposed in year 2014 (Tian *et.al.*, 2014). The literature stated that they also applied the concept of NS which used in NSGAII into Gravitational Search Algorithm (GSA) (Rashedi *et.al.*, 2009; Eldos *et.al.*, 2013; Sabri *et.al.*, 2013). Other than NS, CD also applied to NSGSA-CM. Additionally they added an operator called as chaotic-mutation (CM) in order to prevent the premature convergence. NSGSA-CM also had better strategy of elitism, in which the algorithm selects better solutions of parent and offspring solution based on the NS-ranked population to update the new generation. NSGSA-CM also had a faster evolution process as the author introduced particle memory character and population social information. The Pareto-front set is also come with better diversity and able to handle constraints effectively.

NSGAIII is a latest proposed MO algorithm which also known as the single-reference point based NSGAIII (Yuan Y. *et al*, 2014). It is clearly the extended version of NSGAII. The author K. Deb. (2014) proposed this algorithm stated that NSGAIII able to maintain the best diversity distribution among the population set. This maintenance is aided by supplying and adaptively updating a number of well spread reference points.

From the literature, NSGAIII still depends on the Pareto-dominance to push the solution towards the global Pareto-optimal set. Thus, this lack opens the gate to another improvement. NSGAIII then modified in the same year to form a new θ-NSGAIII which overcome the previous problem of NSGAIII (Bi X. *et. al,* 2017). The author aims to have better trade-off strategy between to retain convergence and diversity. Their method is by adopting a new concept that they introduced which is θ-dominance. θ-dominance is an adaptive mathematical method which to ensure the Pareto-front convergence and diversity. This θ-NSGAIII significantly better that original NSGAIII. Deb (2014) also extends the version of NSGAIII. They proposed U-NSGAIII also to improve the first version of NSGAIII (Seada *et.al.*, 2014).

Multi-objective Genetic Algorithm (MOGA) at some times have become popular in a wide variation of application domains (Murata *et.al.*, 1995). MOGA employed the concept of niching and dominance along with the rank-based fitness assignment. POS provided by MOGA was categorized in group then will be assigned as same ranks in each group. For other groups, POS which was dominated by current group was assigned next ranked. The author dynamically uses updated sharing to maintain the distribution of diversity. The drawbacks from this method is it converges slowly that prevent the algorithm from finding the optimum Pareto-front (Engen, 2010).

Another version of EA is Vector Evaluated Genetic Algorithm (VEGA) (Schaffer, 1985). A modified selection process to cope with several evaluation criteria was adopted to VEGA (Bechikh *et.al.*, 2011). The author, Schaffer proposed to divide the whole population into groups in equal number of objectives. That means, the selection procedure in this small group depends on the single objective. The author also limits the mating operator in order to help limited combinations of individual solutions in the same group. To compare the POS, pairwise comparison is applied. POS are plotted after setup iteration is complete. However, the drawback of this algorithm is it prevents to determine the location of the Pareto-front. The algorithm also lacks in its strategy to select the individuals, as this strategy provide only the better solution in one objective. Hence this led the algorithm to converge to individually best solution only.

The extension of GA then furthered in year 2000 (Corne *et.al.*, 2000). They proposed Pareto Envelope-based Selection Algorithm (PESA) which introduced a new method of small internal population and large external population. They maintained the

concept of grid division in PAES to maintain the distribution of POS. The selection process performed by CD operator. The external population plays the great role in order to maintain the diversity. After a while, author revised PESA to PESAII in year 2001 (Corne *et.al.*, 2001). PESAII differs with PESA in term of its selection procedure. PESAII used region-based selection. The author stated that the concept of an individual is changed to the concept of hyper-box. Then, the individuals are selected from these hyper-boxes randomly. The PESAII was upgraded due to the computation cost in PESA.

As the GA superior to solve optimization problem, the researchers also modified and introduced another version of GA called Micro Genetic Algorithm (MGA) in year 2002 (Chakravarty *et.al.*, 2002). This version of GA is operating with a small population and reinitialization process. MGA generates random population, which then saved into two proportions named replaceable and non-replaceable portions. The replaceable portion is updated every iteration while non-replaceable portion was remained constant throughout the iterations. The author set the population are randomly taken from both of the portion. At the end of each cycle, two POS members from both portions are selected to compare it with the content of external memory. By this operation, all dominated solutions are removed. For elitism strategy, the author lined up with three forms. First form is retaining the POS found within the internal iteration of the algorithm, second by using a replaceable portion of the memory which its contents are partly restored and third is by replacing the population by nominal solutions created. From MGA, author extended it to Archived-Based Micro Genetic Algorithm (AMGA) and its improved version AMGA2. AMGA used genetic operator such as mutation and crossover to create solutions (Tiwari *et.al.*, 2008). For selection, AMGA uses a two-tier fitness assignment mechanism, the primary fitness is the domination-based level rank and the secondary based on the diversity of the solution. AMGA generated small number of new solutions per iteration. This small number of solutions called as micro-GA (MGA). The MGA helped to reduce the number of function evaluation by minimizing exploration of less promising search regions and direction. The use of external memory is maintained by AMGA in order to keep the best solution found. External archive also helped AMGA to provide a wide range of POS. It also provides its flight experience during the operation. Later, the study of AMGA formed AMGA2. AMGA2 was designed to obtain fast and reliable convergence on a wide variation of MO problem (Tiwari *et.al.*, 2011). The author

modified diversity assessment techniques in AMGA and genetic variation operators were proposed. The benchmark test showed AMGA2 improved a lot compared to AMGA.

In addition, there are also literatures that used some concept in MOEA. For example, an MO algorithm named Hybrid Self-Adaptive Algorithm (HSCDA) for community detection in complex networks was developed (Xu *et.al.*, 2015). The author adopted both concept mutation and crossover. Both mutation and crossover were modified. In this paper, they hybrid three new-designed of crossover and two new-designed mutation into a strategy pool, which based on the self-adaptive learning framework. This concept of HSCDA was used to develop an MO algorithm named Multi-objective Community Detection Algorithm (MCDA) (Deng *et.al.*, 2015). MCDA based on the kernel k-means (KKM) and ratio-cut (RC) objective function. RC and KKM are the combination of mathematical method in which introduced into objective function to promote a group of new detection method based on the multi-objective optimization. These two combined methods also used to develop several MO algorithms for community detection such as MOCD, MOEA/D-net, and MOGA-net. These algorithms are interesting in terms of how they find the Pareto-front. MOGA-net for example is an algorithm which employs the community fitness (CF) and community score (CS) as two objectives to be simultaneously optimized. PESAII in the meanwhile is also adopted in MOCD to optimize the objective between intra- and inter- functions. It also employs the two methods of namely max Q and max D to select a suitable solution from Pareto-front. There is also MOEA\D-net which originated from MOEA\D to optimize the negative RC to find dominated solution. This research on modularity-based intelligent algorithm really attracts the researcher's community.

2.4.1 Classification of PoM Approaches

From the literature review, the conclusion can be made that there is still no research to develop MO-type of SDA. The development on SDA only take part in term of improvement in its local optima problem and improves its functions in engineering applications. This gap is yet to be filled with the development of the MO-type SDA. As the performance of SDA is good in solving SO problems, the modification to SDA needs to be done in order to justify the SDA, whether it can handle MO problems. There are many types of MO algorithm that adopting some method to find the POS. From the

findings, there are four methods used by researchers to turn a SO algorithm into MO algorithm. The Table 2.1 below shows these four methods.

Table 2.1 Summary of methods in Posteriori technique.

Method 1	Method 2	Method 3	Method 4
Non-dominated sorting and crowding distance	Archive method	External memory	Mathematical novel adaptive method
The most applicable method used by researchers to develop MO-type algorithm. Most reliable to provide the best accuracy and diversity of Pareto-front set. However, the computation time for this method is quiet longer that 2nd and third method.	The method is one of the simplest methods to find POS for Pareto-front set. Also, able to provide good accuracy and diversity of Pareto-front set.	The memory used to store main population externally. This method only take part in fitness computation. However, the size may increase very large as the iteration continued. A limit needs to be set in order to reduce computation cost.	A complex mathematical equation should be derived to use this method. The mathematical complexity increases the time to compute and increase cost. Not suggested to all application, may the most applicable to solve some applications and problems.
NSGA/NSGAII/NSGAIII, NSGSA-CM, MOWCA	MOPSO, MOGA, NPGA, NPGAII, PAES	SPEA, SPEA2, AMGA, AMGA2	MOBA, MODE, VEGA

For this thesis, the first two methods extracted from various type of MO algorithms are highly recommended to be adopted to SDA. From the study, NS method are the most superior method that adopted to many SO-type algorithms. This is due to the best of CD, which is a mathematical-approach to provide a good distribution of solution set along the Pareto-front. This operator does not feature in any of another three methods listed. Rather than the advantages in the diversity distribution, this method prove able to plot a high accuracy Pareto-front. Another method recommended is AM. From the structure, AM is simpler compared to NS. This method has been used in MOPSO, MOGA, NPGA, NPGAII and PAES. After these algorithms was experimented, they also provenly found as a superior method to solve MO problem. AM is described as a faster strategy to provide a good Pareto-front. In term of diversity and accuracy, the method can be also concluded as superior. For some explanation, the concept of the external memory (3rd method) also applied in AM. The difference between these two techniques are, AM consist of an additional element, which is known as grid-division strategy. The grid

division strategy operator makes lesser computation cost, as the searching will be conducted in the most possible area in the searching space. Last but not least, as the fourth method more complex than others, thus the adoption of this method will be less considered.

2.5 Spiral Dynamic Algorithm

2.5.1 Introduction

This nature-inspired algorithm was introduced in 2011 (Tamura *et.al.*, 2011a). The author inspires the algorithm based on the spiral phenomena which found mostly on earth and galaxies. This method inspired because of the movement of the particles in spiral steps generates logarithmic spirals seems to have a great strategy of solution searching which described as "*diversification in the first half and intensification the second half*" (Nasir *et.al.*, 2012). From the structure of this algorithm, it can be said as simple in design and program. These features make SDA only require a low computation cost to solve SO problems. Additionally, only few parameters that need to be initialized at the beginning. These parameters also make it as a user-friendly algorithm and easy to apply in order to solve the real-world problem. Figure 2.3 shows the strength of SDA.

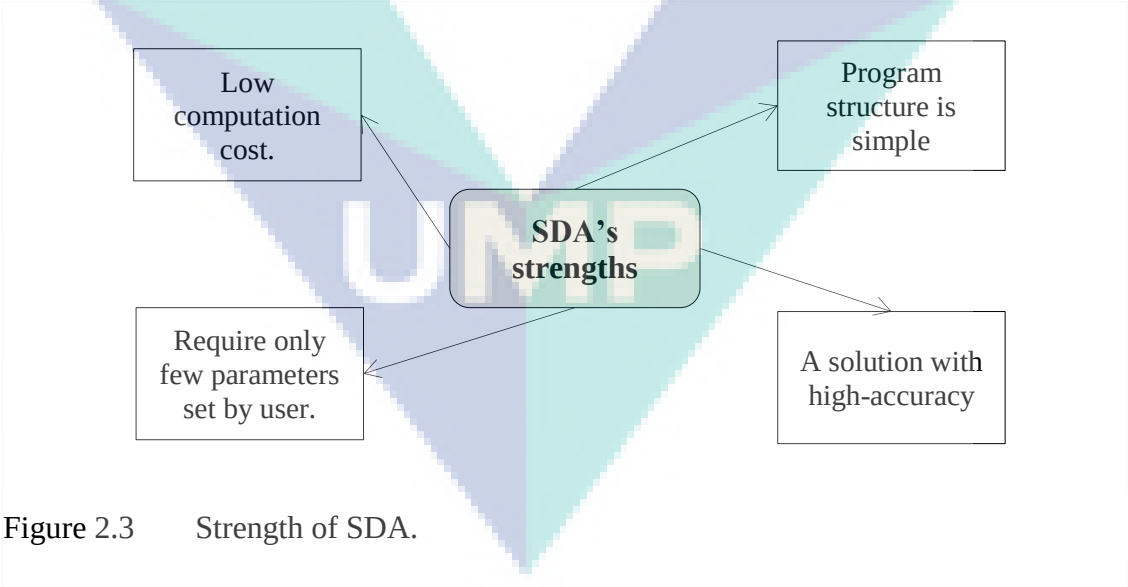

Figure 2.3 Strength of SDA.

The algorithm has strength features in diversification and intensification at the early and later phases of the search trajectory. Recall that the diversification is the ability of the algorithm to explore the entire search space in order to find the solution, while intensification is the ability of the algorithm to exploit found solution in order to converge

towards the global best location (Lakshika, 2014; Tian *et.al.*, 2016). SDA has combination of these features. This strategies lead SDA to search solution with more efficient and provide better solution with high accuracy. This type of algorithm is a derivative-free or non-gradient based optimization algorithm. The algorithm also has a simple structure that has good exploitation and high-speed computing time.

Generally, at the beginning, all agents will be spread onto the search space. These agents will diversely explore the whole search area. At the latter phase, these agents will be intensely moved within a small area of the search space. In counting the spiral steps, the agents will move while the step size gradually decreases as the agents move towards the spiral centre simultaneously. The agents at the first place will be spread and begin to move at the outermost layer toward the centre of the spiral. By using this strategy, the searching agents could achieve global optimum points for various of problem landscape in the search space even it is located in isolated area. Other than that, the spiral step in every iteration is guided by the best cost value from the previous iteration. For the sake of this strategy, the algorithm has a relatively faster convergence speed. To the whole about SDA, it has a simple structured programming which can lead it to provide a solution in a quick time.

The author furthered the study to extend the version of SDA in order to handle more than 2-dimensional continuous optimization problem (Tamura *et.al.*, 2011b). In the paper published in 2011, the author shows how to construct an n-dimensional spiral model. The concept used is rotation matrices in n-dimensional space. This derivation of rotation of matrices are represent the number of dimensions of the problem. Hence the n-dimensional rotation matrices constructed based on these numbers. The advantage of this method is it makes SDA able to deal with n-dimensional problems. The analysis data shows that this method provide solution than the solution provided by PSO and DE (Kennedy *et.al.*, 1995; Chasnov, 2014). As part of conclusion, this method clearly a big improvement for SDA. After that, the author also test the stability of the user-defined SDA parameter (Tamura *et.al.*, 2013). They found that these two parameters, which are convergence rate and rotation rate affect the search performance. They try to proposed the method and focus on the convergence rate, which apply its effectiveness of setting method from analysing stability of dynamics equilibrium point of spiral model.

The extension of SDA also found published by Nasir (2012). This paper presents improved version of SDA called Adaptive Spiral Dynamics Algorithm (ASDA), which adapted with mathematical model equation. The author found that the SDA lack during the search process due to incorporation of single radius value. They apply the novel mathematical equation alongside with non-mathematical fuzzy logic strategy. This both method important to establish the relation between fitness value and spiral step radius well. In this paper, they introduced four approaches of adaptable step size of SDA. The first method is by applying a non-mathematical fuzzy logic and the rest is by utilizing the novel mathematical equation based on linear, quadratic and exponential function. From the study, it can be concluded that these adaptive SDAs are better than the previous SDA. It provides higher speed of convergence and higher accuracy. Moreover, these version ASDA are still retained the simple structure of SDA, which mean its complexity is not changed.

Another one improved version of SDA is exponential-based SDA (ESDA) (Nasir *et.al.*, 2015). The author used this ESDA in order to model a flexible manipulator system. In this ESDA, the author has improved the feature of spiral movement throughout the search by adapting mathematical method to vary both of the radius and angle of the spiral model. An exponential-based radius and exponential-based angle are adopted into SDA. To test the performance of the ESDA, the author applied it to solve dynamic modelling problem of a single-link flexible manipulator. ESDA is found improved significantly from original SDA and show a good dynamic behaviour of the system. To summarize ESDA, it provides better model of the system and contain less error compared to SDA.

SDA also applied to solve a Combined Economic and Emission Dispatch (CEED) problem (Benasla *et.al.*, 2014). In this problem, SDA managed to minimize fuel cost and emission levels. It scheduled the generators, while trading-off the requirement of load demand and operational constraint. Both of the objective is combined into single objective function using price penalty factor. The author used SDA on three test systems with different number of generating units. Each of them has their own constraints and various cost curve nature. From the result, SDA solve the problem by minimizing a good performance of the CEED problems.

Another foundation, SDA also used to evaluate human health effects of using computer-aided workstation (Khan *et.al.*, 2013). Human and safety (HS) in this journal

are related to occupational ergonomics and job satisfaction. To calculate the HS risk, index a neural-swarm spiral-dynamic search (NSSS) optimization-based algorithm has been employed. NSSS is an example of multi-disciplinary optimization involving SDA, which a form a hybrid algorithm of SDA with neural-swarm-based HS risk assessment model which use artificial neural network (ANN). ANN is a set of processing elements, containing neurons or nodes that interconnecting with each other. NSSS interpret the weight matrices of the ANNs as solutions, weights, and to change the weights. This is achieved by using iterative spiral movement to find the better solution. The author set four human risk indexes denoted as I_{HS} which are low, moderate, high and extremely high. An advantage of this NSSS is it is easy to use and the user can quickly point out individual employee specific HS risk.

From the study, SDA has unbalance exploration and exploitation strategies due to the spiral movement. To be clear, SDA has a simple structure that has good exploitation but not better in exploration. This lead the solution to get trapped into local optima point, a situation which no modification made to the current best result that lead to produce better solution (Michalewicz *et.al.*, 2004). The measurement of 'better' depends on the performance of the solution with respect to the single objective being optimized (Knowles *et.al.*, 2004). As result, the algorithm might produce a solution which has low accuracy. Despite of its great abilities, SDA remains one of the metaheuristic algorithms which is not extended to become a multiple-conflicting objectives problem solver so far as there is no such literature found. Among conventional study on this algorithm, there is no version of SDA dealings with MO problem plus there is only a limited number of publications in literature on improving its performance in term of accuracy and convergence speed. Although it had a little weakness in its strategy, research of SDA is still far from maturity. A lot of studies may be required because it is relatively such a new metaheuristic algorithm that has good potential to be improved. Therefore, to know the real ability of this algorithm, this SO SDA could be expanded to its MO algorithm in order to deal with MO problem, which is more complex and more realistic to the various of current applications.

For conclusion, there are a lot of version SO-type SDA developed by the researchers nowadays. Thus, there is still a gap that need to be fulfilled by developing new version of MO-type of SDA.

2.5.2 Program Structure

Essentially to emphasize that to modify SO-type SDA into MO-type algorithms, the fundamental of SDA is really important to be technically explored deeper. The elements of SDA will be explained in detail in this section. SDA provides dynamics step size for each agent to move from one point to another, in spiral model (Tamura *et. al.*, 2016). Spiral model is represented by an equation. This equation is the most important in SDA because this equation will move all the search agents a step ahead in spiral steps. The previous mentioned features of diversification and intensification are generated by this equation. This equation also consists of crucial parameters that need to be defined at first place. These parameters affect the performance of the algorithm, in term of its speed of convergence to find the solution and how accurate the solution is. The spiral model equation of SDA is defined in Equation 2.6.

$$x(k+1) = S_n(r,\theta)x(k) - (S_n(r,\theta) - I_n)x^* \qquad 2.6$$

From the equation 2.6, x^* is the centre of the spiral, I_n is the matrix identity, $x(k)$ is location of a point at k^{th} iteration, k is the iteration number, r is radius of spiral, θ is the angular displacement and $S_n(r,\theta)$ is a square matrix formulated based on radius, r and rotation matrix with $n \times n$ dimension. $S_n(r,\theta)$ is represented in Equation 2.7.

$$S_n(r,\theta) = rR^{(n)}(\theta_{1,2}, \theta_{1,3}, \ldots., \theta_{n,n-1}) \qquad 2.7$$

From the equation 2.7, $R^{(n)}(\theta_{1,2}, \theta_{1,3}, \ldots., \theta_{(n,n-1)}$ is $n \times n$ dimensional of rotational matrix. For clarification, $R^{(n)}(\theta_{1,2}, \theta_{1,3}, \ldots., \theta_{n,n-1})$ also can be representing Equation 2.8.

$$R^{(n)}(\theta_{1,2}, \theta_{1,3}, \ldots., \theta_{n,n-1}) = \prod_{i=1}^{n-1}\left(\prod_{j=1}^{i} R^{(n)}_{n-i,n+1-j}\left(\theta_{n-1,n+1-j}\right)\right) \qquad 2.8$$

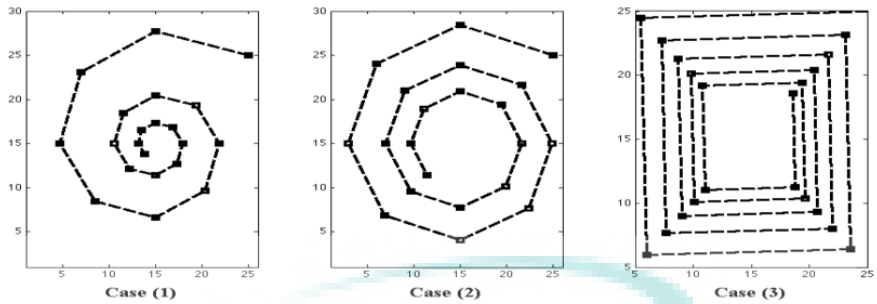

Figure 2.4 Different value of r and θ resulted the shape of the spiral step.

With respect to these equations, types of the spiral model with different value of radius, r and angular displacement, θ are illustrated in Figure 2.5.

Denote that when different radii r and angular displacements, θ are used, then the shape of the spiral is also different. This shows how both parameters will affect the search. The performance of SDA will be depending on these parameters. For instance, every spiral model has its own centre and in SDA, the centre, x^* will be chosen according to best ranked agent in the current population and it changes in every iteration. If the top-ranked solution found in the next iteration is better than the top-ranked found in the current iteration, the new x^* will be defined.

Step 0: Preparation
Select number of search agents, $m \geq 2$, angular displacement, $0 \leq \theta < 2\pi$, radius of spiral, $0 \leq r < 1$ in $S(r, \theta)$ and maximum number of iterations, k_{max}. Set $k = 0$.

Step 1: Initialization
Randomly spread search agents $x_i(0) \in R^n,\ i = 1,2,3, \dots, m$ in the feasible region. Set centre of spiral, x^* as $x^* = x_{i_g}(0),\ i_g = \arg\min_i f\left(x_i(\dot{0})\right),\ i = 1,2,3, \dots, m.$

Step 2: Updating position of search agents
Update position using spiral equation:

$$x_i(k+1) = S_n(r, \theta)x_i(k) - S_n(r, \theta) - I_n)x^*$$

Step 3: Updating centre of spiral
The centre of spiral is updated after positioning the agents by $x^* = x_{i_g}(k+1)$, where $i_g = \arg\min_i f\left(x_i(k+1)\right),\ i = 1, 2, 3, \dots, m.$

Step 4: Checking termination criterion
Terminate iteration if $k = k_{max}$. If else set $k = k + 1$ and repeat step 2.

Figure 2.5 Pseudocode of SDA.

The pseudocode of SDA is shown in Figure 2.5. The sequence starts with selecting the number of search agents, m. The number of agents will affect the accuracy of the solution. The large number of agents provides more accurate of solution while small number of m will reduce the accuracy. However, if the number is too large, then it will affect the computation time. Hence, the enough number of search agents is required. In the same stage, the user determines the angular displacement, θ and radius of spiral, r. These parameter values selections are also important. As mentioned before, they will affect the performance of the algorithm. After the preparation, m of search agents will be randomly spread out in the search space or called the feasible region. These agents will be put onto these spaces randomly in term of position or point, between the specific ranges of the problems. These points then will be evaluated on the problem function. The best cost value among these positions will be set as the centre, x^*. All of the points will move towards this x^* as in Step 2. These points further will move in spiral step by using the spiral model equation. For a clearer vision, this equation will be used to move all of the points towards the spiral centre in spiral step. This explains why large m will make the algorithm computes slower because it needs to calculate a lot of new positions for each agent. These new points then will be again evaluated on the problem function and the new x^* is determined by the best cost value from the computation. This sequence will be repeated till k_{max} is achieved and if not, the procedure will be restarted from Step 2.

2.6 Genetic Algorithm and Its Variation

2.6.1 Introduction

Genetic algorithm (GA) is designed to simulate the biological process inspired by evolutionary theorem of Charles Darwin. GA at the early phase was introduced by James Holland in 1960 however become attraction to many researchers in late 1990. It searches the solution through the use of simulated evolution, by adopting the concept of survival of the fittest (Bodenhofer, 2004; Sivanandam, S.N., *et.al.*, 2008; Carr, 2014; G. Kumar, 2014; Kramer, 2017). The terminology also taken from the biology. For instance, the fittest member in the simulated population tend to survive, reproduce and continue the to the next generation, then improving for each generation. However, some of the inferior member also able to survive, by a little chance to reproduce. This GA procedure is much simpler compared than its real biological counterparts, which contain the components of

five including 1) the fitness function of the MO problems; 2) a population of chromosomes; 3) selection of the fittest chromosomes; 4) crossover to produce next generation and 5) mutation to random member in population. The word "fitness" used widely in extended GAs is extracted from the evolutionary theory. This word was chosen to represent how fit the potential solution is. The chromosome represents the candidate solution. These chromosomes which actually is numerical value is the one that the GA trying to optimize.

GA has been widely used to solve many linear and non-linear problems by exploring the all regions of the state space through mutation, crossover, and a selection operation (Michalewicz, 1996; Houck *et.al.*, 2008). To solve different problems, researchers have modified many versions of GA.

Adaptive GA (AGA) is the one of the enhanced versions of GA. In AGA, the parameters are varied throughout the searching process (C.L. *et.al.*, 1993). Those parameters are including population size, rate of crossover and rate of mutation. The author also made variant of several AGA by varying the parameter such as the mutation rate will be changes accordingly to the changes in the population. As long the population do not improve, a higher mutation rate is chosen. The AGA also will be acting vice-versa if the population is improved.

Another work improving GA is done in 2017 (Nasir *et.al.*, 2017). The author combined GA with SDA to form a Hybrid-Spiral-Genetic algorithm (HSGA). From this paper, the exploration and exploitation strategy in GA are improved by adopting spiral model from SDA. After tested with several benchmark functions, HSGA is clearly outperform both of SDA and GA at a significant level of improvement.

One of the improved GA was developed to optimize a multi-runway schedule of airport (Zhou *et.al.*, 2015). The author used sliding time window algorithm (STWA) to improved GA. STWA is an algorithm developed focused to solve flight landing schedule. The current STWA faces problem of ineffectiveness in schedule the flights and take-offs, delays and landing. In this paper, GA was improved in order to adopted the conflicting problems. They applied Grefenstette coding method to produce efficient chromosomes. So, these chromosomes are representing the sequence of the flight. At the same time, Grefenstette also reduced the complexity of the algorithm. By hybridizing GA and

STWA, the author found that the runway throughput was increased by 13% and delay cost was reduced by 61%. The controller of the runway also significantly reduced in term of its workload compared to traditional method of "first come first serve". Although complexity of the proposed algorithm was increased, the algorithm managed to reduce time complexity and improves real-time and work-efficiency significantly. This GA-STWA significantly succeed to make better air traffic control.

2.6.2 Program Structure (Carr, 2014)

At the beginning, it is important to emphasize that mutation and crossover operators are involved in GA. The mutation procedure and the crossover operator used can be described in Figure 2.6 and Figure 2.7 respectively. At the meanwhile, Figure 2.8 shows the pseudocode of GA. GA at the beginning will initialize a Population (pop) with desired number of chromosomes, denoted as n_{pop}. These chromosomes then will be tested with the fitness function, in order to obtain the information its fitness value. The selection will be furthered operates to find the fittest solution among the members of the population. The members which are fitter have greater chances to survive and reproduce. At the next phase, the selected chromosomes will undergo crossover in Figure 2.6. The crossover operator simulates the biological crossing over and they will be combined in cell meiosis. This process will produce two chromosomes, which swapped from subsequent of two previous parent selected chromosomes. As shown in Figure 2.7, the mutation operator will take the current found population as their subject. This operator is defined to create a new position of agents in the search space, according to lower and upper boundary, which computed based on the mutation rate initially set by the user. The mutation operator will randomly flip the individual properties by change the bit '0' to '1' and vice-versa. As the stopping criterion is still not met, this cycle will be repeated until the initial population is completely replaced with the fittest solution found.

Table 2.2 Symbols for crossover operator.

Symbols	Description
x_i	i^{th} number of individuals
x_j	j^{th} number of individuals
y_i	New position for i^{th} agent
y_j	New position for j^{th} agent
α	Rate of crossover

Step 0: Preparation
Set rate of crossover, select specific two individuals in current population, i^{th} and j^{th} of in individual in pop and named x_i and x_j. Take the position of both agents.

Step 1: Initialization
Define the size of position agent x_i. Random a value between range of this size.

Step 2: Operate crossover
Compute the new positions for both i^{th} and j^{th} by using crossover equation.

$$y_i = \alpha \times x_i + (1 - \alpha) \times x_j$$
$$y_j = \alpha \times x_j + (1 - \alpha) \times x_i$$

Step 3: Store new position, x_i and x_j.

$$x_i = y_i \text{ and } x_j = y_j$$

Step 4: Return x_{new}

$$x_{new} = \begin{cases} x_i \\ x_j \end{cases}$$

Figure 2.6 Crossover Operator

Step 0: Preparation
Select a constant value for rate of mutation, $mutrate$.

Step 1: Initialization
Take current pop agents, identify the current number of NDS in pop. Choose randomly an agent in pop, called $j = rand(n)$.

Step 2: Calculate rate of mutation
Compute the rate of mutation, in between the range of MOP.

Step 3: Mutate agents.
Set the lower and upper boundaries for j, and create a new random position of agents, x_{new}

Figure 2.7 Mutation Operator

Step 0: Preparation
Choose total number of chromosomes in the population, n_{pop}, mutation rate and crossover rate.

Step 1: Initialization
Create chromosomes in a population, pop. Evaluate the chromosomes using fitness cost function.

$$Fitness\ cost = f_n\big(x_i(k+1)\big)$$

Step 2: Select the best chromosomes with best fitness value for reproduction.
The best n_{pop} of chromosomes with best fitness value will be selected to reproduce by next breeding operators.

$$Best\ chromosome = \min(fitness\ cost)$$

Step 3: Breed new children by crossover and mutation.
Apply crossover and mutation procedures as stated in Figure 2.6 and 2.7 to double the size of n_{pop} by producing children.

Step 4: Evaluate fitness of new chromosomes.
Combination of parents and children will be evaluated to know their individual fitness.

$$Fitness\ cost = f_n\big(x_i(k+1)\big)$$

Step 5: Replace least-fit chromosomes
New chromosomes with better fitness will replace least-fit chromosomes.

Step 6: Check termination criterion
Check termination criterion, if $k = k_{max}$, then stop. If not, then repreat Step 2.

Figure 2.8 Pseudocode of GA.

2.7 Fast-elitist Non-dominated Sorting Genetic Algorithm

2.7.1 Introduction (Barthelemy *et.al.*, 1993)

To adopt the NS methodology concept which originated from NSGAII into SO-type SDA, the fundamental of NSGAII is really important to be explored further. Figure 2.9 shows the elements of NS techniques. These elements will be explained in detail in this section.

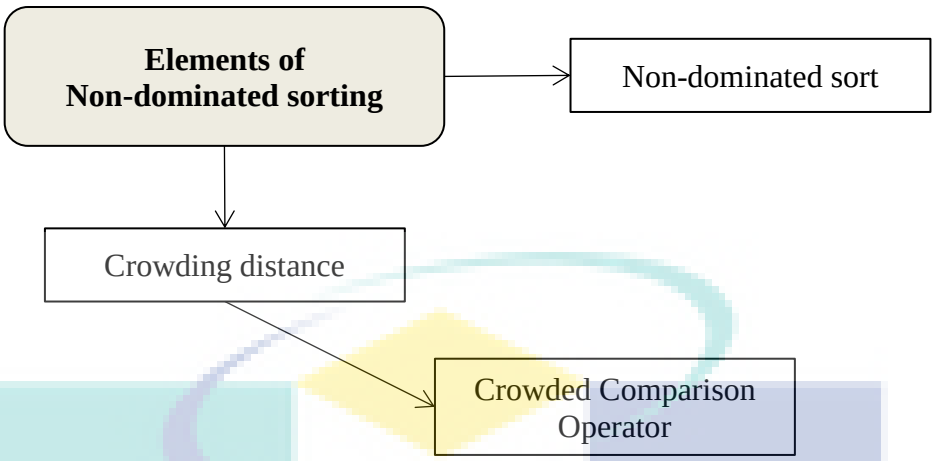

Figure 2.9 Elements in NS-approach

2.7.2 Flowchart NSGAII

Figure 2.10 shows the flowchart for NSGAII. At the beginning, the algorithm will initialize a size population of n_{pop} and will be evaluated based on MOP. After the evaluation, all individuals will be sorted to determine their domination. This is where the Pareto-optimal concept is applied. At this phase also, the individuals will be arranged in ascending order, based on their best fitness values. This initial population is recognized as parent, in which this size of population furthered to increase by producing children. In this case, NSGAII will use the GA operator to produce those children. There are three components of GA which are binary tournament selection, mutation and crossover. Binary tournament selection is a function to select individuals which are the fittest. These selected individuals will be mutated and crossover to produce new generation. The number of the desired offspring or children is defined by user at the beginning. Producing more children will create more chances to produce better solution however increase the computation time.

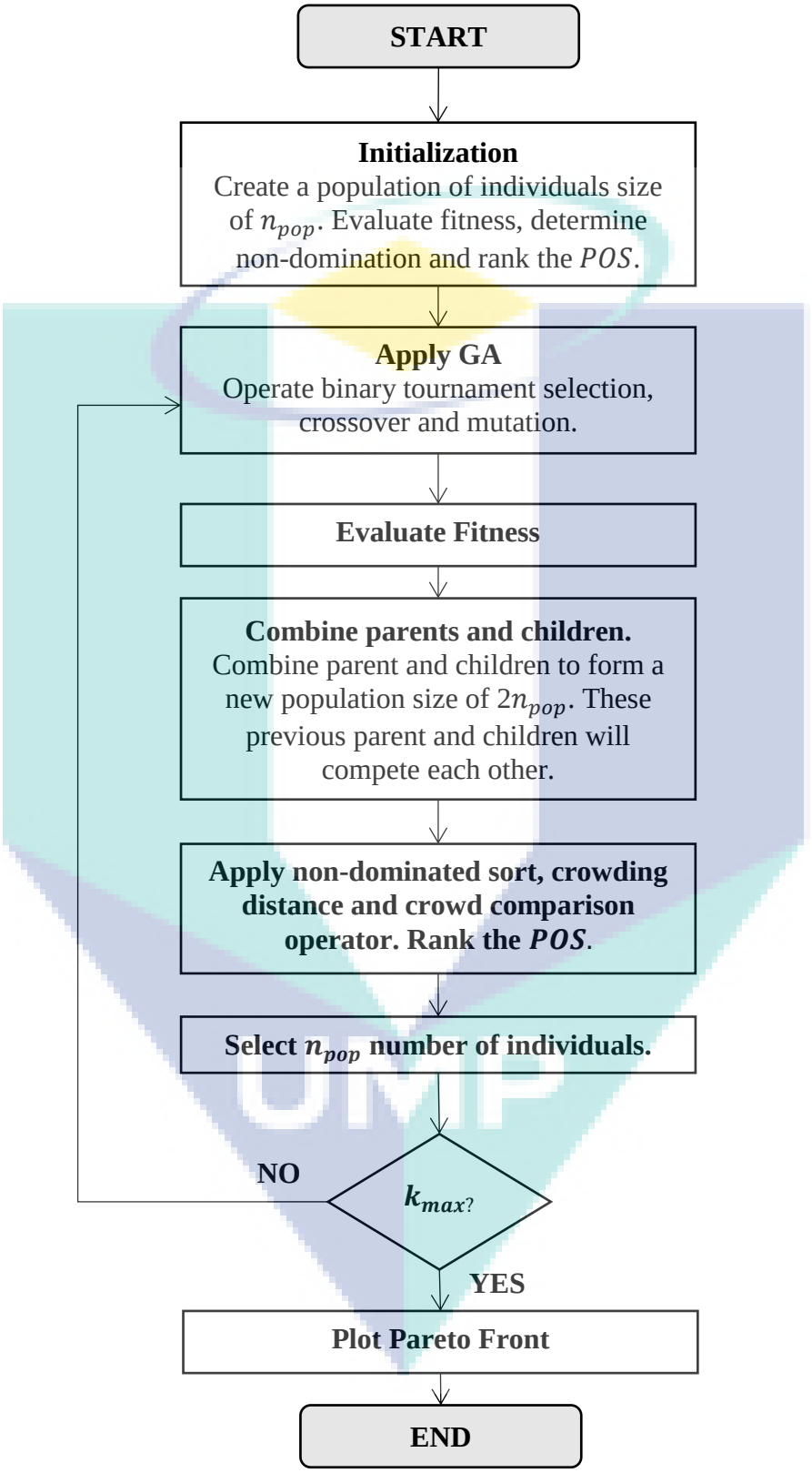

Figure 2.10 Flowchart of NSGAII.

The process is continued in order to evaluate the entire population based on the MO problem. In this phase, all individuals from parent and children populations are evaluated. Then the information of their fitness will be stored in each individual characteristic.

Next, both parent and children population which were combined will be sorted using non-dominated-sort operator. By this process, the individuals will be again separated and stored according to their non-domination level. Then to preserve the diversity, the sorted population will undergo both CD calculation and CCO.

The last step is to discard the excessive individuals in the population and let the required size of population remains. The crowded comparison operator (CCO) will be applied to select the best desired number of individuals to continue and survive for next iteration. This process will be continued till the maximum iteration is achieved.

Once the algorithm achieved the maximum iteration, the last population will be used to plot the Pareto-front. Thus, this is the best POS found that plotted along the Pareto-front, which can be used as the best solution.

NS is an organized and systematic strategy to find the POS. This is shown in Figure 2.11. Important to denote that, the NS method is not able to function well if these three elements are not applied together. Therefore, when NS is used to developed a new version of MO-type SDA, these three elements are included.

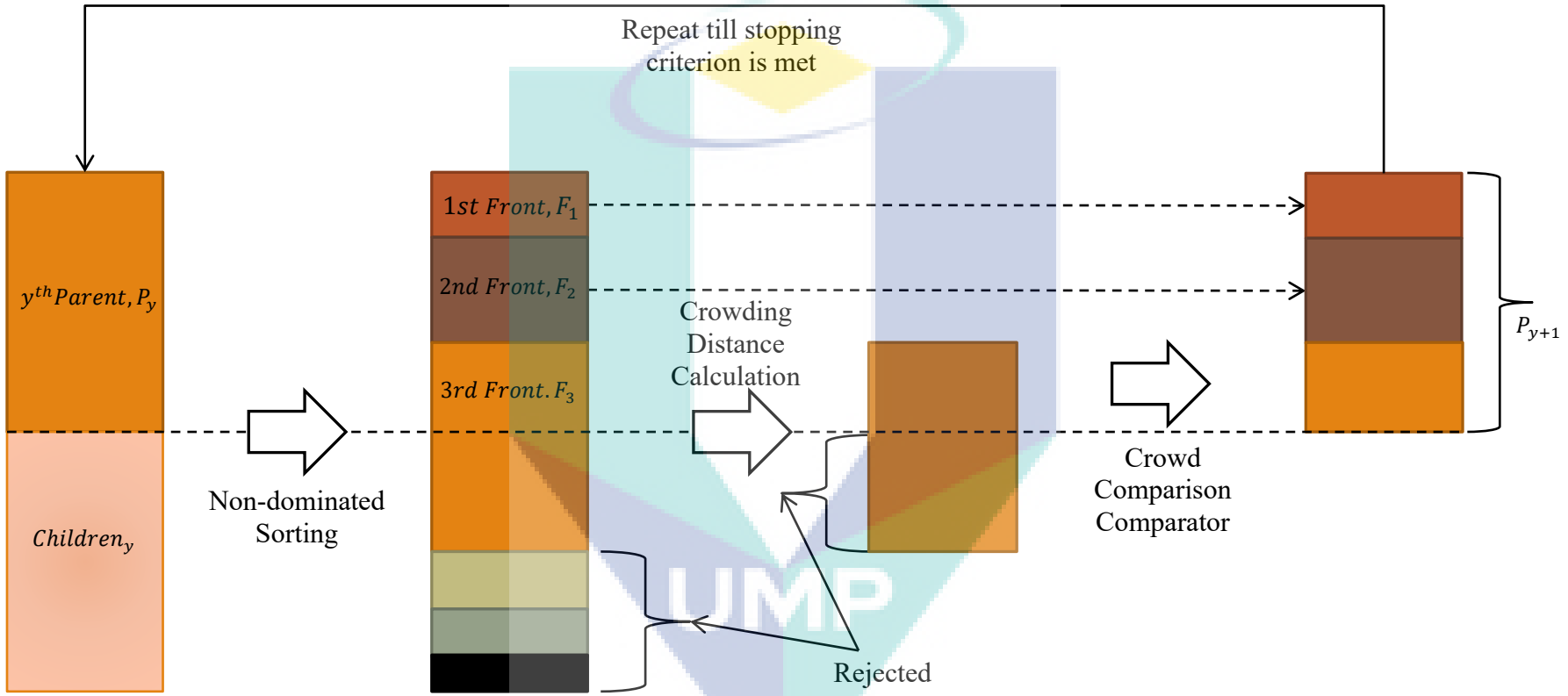

Figure 2.11 Non-dominated sorting procedure illustration.

2.7.3 Non-dominated-sort (Barthelemy *et.al.*, 1993)

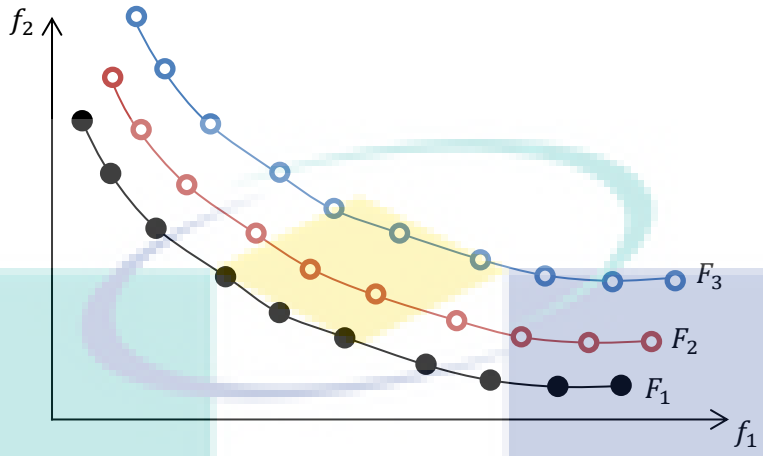

Figure 2.12 Non-domination level (NL). Denote that, $F_1 > F_2 > F_3$.

Non-dominated-sort is a methodology which divides a population of solutions into several non-domination levels (NL).

Let two solutions, y_1 and y_2 from a population size of n_{pop} are taken to make comparison.

Definition 1:
Solution y_1 is considered to dominate solution y_2, if both conditions (a) and (b) are satisfied.

(a) y_1 is no worse than y_2 in all objectives or,
$f_j(y_1) > f_j(y_2)$ for all $j = 1,2,3, \ldots, n_{pop}$
(b) y_1 is strictly better than y_2 in at least one objective or,
$f_j(y_1) < f_j(y_2)$ for all $j = 1,2,3, \ldots, n_{pop}$

Definition 2:
Non-dominated set is a set of all solutions that are not dominated by any member of the solution set.

Figure 2.13 Condition in selecting Pareto-optimal solution (POS).

Figure 2.12 illustrates the concept of NL. This division of NL is based on their dominance relationship. This method provides a good quality of solution in which NL they belong to with respect to each other. However, it becomes time-consuming when the number of objective and population size is increased. In other words, in non-dominated-sort, each solution must be compared to each other within the population by using the

condition in Figure 2.13. This is to sort the population, pop in size of n_{pop} according to the non-domination level.

For the initial clarification, to describe the performance or complexity of an algorithm in computer science, big "O" notation is used. Big "O" specifically describes the worst-case scenario and can be used to describe the execution time required. Other than that, this notation also describes the space used for example in memory of disk by an algorithm.

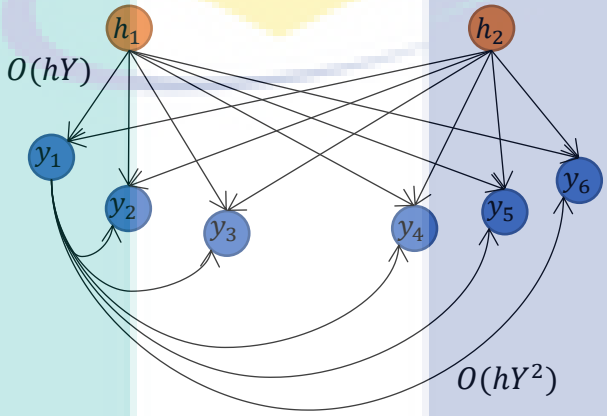

Figure 2.14 $O(hY)$ and $O(hY^2)$ complexities illustration.

The comparison requires $O(hY)$ of complexity for each solution, which h is the number of objectives and $Y = y_1, y_2, y_3, \dots, y_{n_{pop}}$. Figure 2.14 shows the illustration of this complexity. In this figure, there are two objective functions, h_1 and h_2. The individuals, $y_1 - y_6$ will be evaluated on these objective functions. The complexity then will become $O(hY^2)$ when the process is continued to find the first NDS as shown in Figure 2.14. The complexity increases because the comparison of an individual to other each individual take place. During this phase, all first POS are found. To find next frontiers, the first POS are temporarily discounted. The procedure then repeated with the same complexity of $O(hY^2)$.

In general, the procedure will initialize the problem-range-based population of search individuals. Fitness or cost value of these searching individuals is then evaluated by recalling the MO problems. Then, the individuals in the population will be sorted based on the NL. POS that have different levels of non-domination will be stored in different subsets. Denote that, a subset is a set of POS in different NL. In this perspective, the method is done by ranking the current non-dominated subset as 0 and then it is

removed from consideration temporarily. Next, the remaining population is evaluated to determine another non-dominated subset. If there are, the subset will be ranked number 1. This sequence will be continuing until the overall population has been ranked. This non-dominated-sort procedure in which applied to a population will return a list of non-dominated fronts, F.

Step 0: Preparation
Calculate the fitness of each individual in n_{pop} size of population.

Step 1: Initialization
Evaluate each evaluating dominate member, p of population, P which has entities of domination count, n_p, and a set of solutions, S_p, which are the set of solution p dominates. Set 1st front domination count, $n_p = 0$ and $S_p = 0$.

Step 2: Determine first domination
Visit each member, q in P. Compare p against each visited member, q. If p dominates q, add q into S_p. Otherwise, if q dominates p then set increment of 1 in n_p.

Step 3: Ranking the solution
Rank 1 for each member p which has $n_p = 0$, and combine them with the current first frontier, F_1.

Step 4: Archive the current front
Store all the NDS in F_1 in storage Q.

Step 5: Determine solution domination of current iteration.
Start front counter, $i = 1$. Evaluate each p in current front, F_i by comparing it with each visited member q of new updated set of population, S_p. For each q in S_p,set decrement of 1 in domination count of q, n_q. If $n_q = 0$, then store q in Q.

Step 6: Checking termination criterion
If $i = k_{max}$, then terminate, otherwise set $i = i + 1$ and repeat step 4.

Figure 2.15 Pseudocode of non-dominated-sort procedure.

Figure 2.15 shows the pseudocode of non-dominated-sort procedure. By this detail procedure, at the early stage of its operation, a subset of population P, denoted as S_p is initialized with all of the individuals that dominated by p is contained in this set. At the same time, operator also initialized the number of individuals that dominated by p, n_p. In the first step in the pseudocode, the evaluation of the individuals begins to take place. All individuals in population will be compared each other's in order to determine the domination. For an individual, p will be compared with another individual, q. At this

stage, if p dominates q, individual q will be included in solution set, S_p. Otherwise, if q dominates p then the domination counter will be increased by 1. Non-dominated-sort will set p as the individual of first frontiers if there is no member dominates it. Non-dominated-sort will then continue its operation with storing the current fronts in a separated list, H. The individual, p in each front counter i, F_i is then again evaluated. The operator will compare the individual, p with q in the solution set, S_p and it will decrease the number of domination count, n_q until it becomes zero. When the n_q is empty, it means that there is no individual in the remaining i^{th} fronts could dominate q. Hence, the ranking is set to increment 1, $q_{rank} = i + 1$. The set with individual q then will be combined with the current front, $\mathrm{H} = \mathrm{H} \cup q$. The front counter is then increased by one and the final set of current fronts, H is now the next front. At the other meaning, the current i^{th} front, F_i is equal to H or $F_i = \mathrm{H}$.

Non-dominated-sort procedure in NSGAII is better compared to the same procedure in NSGA because it utilizes the information about the set that an individual dominates (S_p) and number of individuals that dominate the individuals (n_p).

2.7.4 Crowding Distance

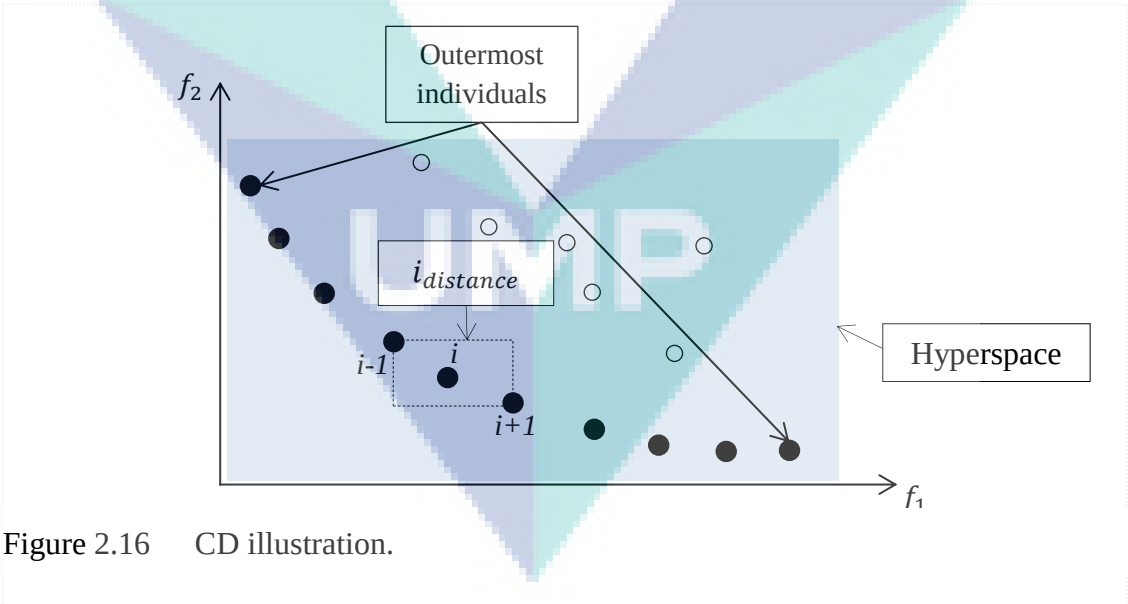

Figure 2.16 CD illustration.

The next procedure in NS is to calculate the density estimation or crowding distance (CD) that surrounding a particular individual in the population. This point refers to the fitness value of an individual. In other words, CD is an operator which used to

define an order among individuals. The way to calculate CD by taking the average distance of two points between an evaluating point along each of objectives, $i_{distance}$.

Figure 2.16 shows the illustration of CD. CD computes $i_{distance}$ which represent the cuboid enclosing one individual (i) without accounting any other points of the population. The idea behind the creation of CD concept is actually to find a Euclidian distance between each individual in a frontier's members. This $i_{distance}$ will operated to serve the estimate of the size of the largest cuboid enclosing an individual, i without considering any other individuals in the population. The front is based on their h objectives in their h dimensional hyperspace, which mean that the CD only operates on the individuals on the same front. Denote that hyperspace is another term for searching space or feasible space. Also, important to stress that, two outermost individuals which are located at right- and left-hand-side of the whole searching space are always selected since the individuals are set with infinite distance assignment.

Step 0: Preparation
Take the information of current frontiers. Determine number of NDS in the current front, I.

Step 1: Initialization
For each iteration, i set $\mathrm{I}(i)_{distance} = 0$.

Step 2: Sort the member according to fitness value
Each member in F_i is sort in ascending order. Set for the first distance, $\mathrm{I}(1)_{distance} = \mathrm{I}(i)_{distance} = \infty$.

Step 3: Calculate crowding distance
Calculate CD for $i = 2\ to\ (\mathrm{I} - 1)$ by using equation

$$\mathrm{I}(i)_{distance.new} = \mathrm{I}(i)_{distance.current} + (\mathrm{I}(i+1).h - \mathrm{I}(i-1).h)$$

Figure 2.17 Pseudocode of CD.

Figure 2.17 shows the pseudocode for CD. Particularly, $\mathrm{I}(i).h$ refers to the h^{th} objective function value of the i^{th} individual in the set I. Denote that, term "*objective function value*" is equal to term "*fitness value*" in NSGAII while $\mathrm{I}_{max} = n_{pop}$. In the figure, the pseudocode of CD starts with taking the information of location of current frontiers. This preparation step also determines the numbers of current POS that found. After that, the algorithm will initialize all distance, $\mathrm{I}(i)_{distance}$ of all these POS by 0. Continuing the operation, all the POS will be sorted in ascending orders based on the

fitness value and non-domination levels (NL). For the first- and last-ranked POS, their initial distances will set by infinity to make them always selected. Next procedure is to calculate the CD by using the stated mathematical formulation. This procedure will be applied to all of POS found in the storage as shown in Figure 2.18. At the end, only N individuals with large $\mathrm{I}(i)_{distance}$ will be selected.

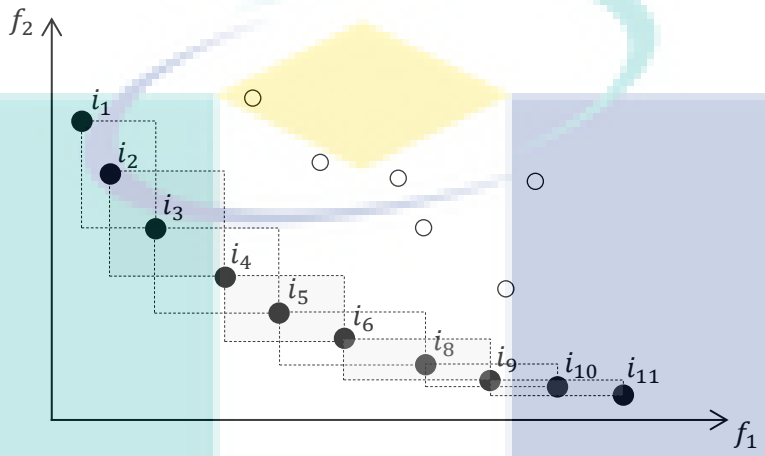

Figure 2.18 CD evaluation on all individuals in the same front.

For instances, to be clearer, the algorithm will generate $i_{distance}$ for each inidviduals as shown in Figure 2.18. i_1 and i_{11} will always be selected because they are the outermost POS in the feasible region. In this figure, to generate $i_{distance}$ for i_2, i_1 and i_3 will be the vertices for the cuboid that enclosed it. This procedure also will be repeated to $i_{distance}$ for i_3 in which i_2 and i_4 will be its vertices. This step will be repeated to all i_3 to i_{11}. As the consequence, individual i_{10} will not be selected because it enclosed in the smallest cuboid. To demolished i_{10}, next operator called CCO is involved. This operator will be explained in the next subsection.

Really important to denote that CD cannot be operated between two individuals in different fronts. It is only applicable front-wise, which means it is only can compare distance between two individuals in same front. To get a better diverse Pareto-front, individuals which have a large CD value are better than ones with a smaller value, if they are at same NL. Other than that, CD with combination of its component Crowded Comparison Operator (CCO) also operates to remove all solutions with worst CD from the storage, when the storage has achieved its limit.

2.7.5 Crowded Comparison Comparator

To obtain a well-diverse Pareto-front, CCO is important. CCO is a part of CD. To operates this CCO, two inputs are needed, which are 1) POS rankings, i_{rank} ; and 2) CD values, $i_{distance}$. These both two values come from both non-dominated sort and CD operators respectively. These two values are also can be defined as the attributes from each individual of the population, which passing through both non-dominated-sort and CD procedure. This procedure is described as in Figure 2.19.

Step 1: Execute partial order
Define partial order, $\geq_n$.
$i \geq_n j$, if $i_{rank} < j_{rank}$ or $i_{rank} = j_{rank}$ and $i_{distance} > j_{distance}$

Figure 2.19 Pseudocode of CCO.

CCO only consists of one step. The step only requires the POS found to satisfy the condition of the defined partial order which denoted with notation ($\geq_n$). From the definition of CCO , when two solutions with different non-domination ranks, the individual which has the lower rank will be selected. In contrast, if these two individuals are in the same front, then the CCO will select the member which located in a space with lesser number of frontiers. This will be decided if the size of cuboid, which bound the points is larger.

2.8 Particle Swarm Optimization and Its Variation

2.8.1 Introduction

Particle swarm optimization (PSO) inspired from the emulation of the group of dynamic behaviour choreography of a bird flock (Kennedy *et.al.*, 1995). This means that every particle in a formed population, all particles affected by the overall groups. Each particle is set with position and velocity. PSO combines the experience of the particles with the experience of the group. The behaviour of each member is affected by either the best local or the best global. The concept can be seen as a distributed behavioural algorithm that performs multi-dimensional search (Coello *et.al.*, 2002). The concept in evolutionary algorithm (EA) which is population and a measure of performance similar to the fitness value also adopted (Coello, 2006). The adjustment to the members also done with crossover operator. At another aspect, PSO allows members to take advantage from

their past experiences. In EA, these current population is the only memory used by members. Similar with GA, PSO also manage to solve linear and non-linear problems (Kennedy *et.al.*, 1995). Based its structure, PSO is really suitable to be modified into MO algorithm and can solve MO problems as it has high speed of convergence when it performs to solve SO problems.

A lot of researchers have developed a numerous number of modifications to PSO in order to solve the non-linear searching problems. One of the version are the orthogonal PSO (OPSO), which adopted simple orthogonal array of Taguchi method (Kuo *et.al.*, 2010). To apply this, the functions are initially defined. These number of objective functions are respectively representing the number of swarms. For example, if three functions are defined, then the orthogonal array will contain three factors and three levels. The optimal solution is able to be obtained by comparing the values in this array. To validate this method, the author applied it to an axial flux motor system and they managed to maximise the motor RPM alongside with its torque better.

2.8.2 Program Structure

Step 0: Preparation
Set two acceleration constants, c_1 and c_2. Set desired maximum iteration, k_{max}.

Step 1: Initialization
Initialize the population, pop with specific number of particles, n and randomized position, x.

Step 2: Calculate the fitness value
All particles are evaluated to know its fitness by using equation below.

$$Fitness\ cost = f_n\big(x_i(k+1)\big)$$

Set best-fitness value gained by each particle as their personal-best ($pBest$) fitness value.

Step 3: Choose a particle with best-fitness.
A particle with best fitness value will be selected as Global-best ($gBest$) particle.

Step 4: Calculate new velocity
Each particle will have its velocity value using the following equation.

$$v_{new} = v_{current}\ c_1 \times rand\ \times (pBest - x) + c_2 \times rand\ \times (gBest - x)$$

Step 5: Update position, x
Particles position are updated using the equation below.

$$p_{new} = p_{current} + v_{new}$$

Step 6: Check termination criterion
Check termination criterion, if $k = k_{max}$, then stop. If not, then repreat Step 2.

Figure 2.20 Pseudocode of PSO.

Figure 2.20 shows the procedures that take place in PSO. It consists of six steps in total. To generalize the procedure in PSO, after the searching begins, the algorithm will set the initial solution (Ibrahim *et.al.*, 2015). While the algorithm iterates, every particle in population will be updated with a new value coming from the evaluation of group particles and individuals' particle. The accuracy of the solution depends on the fitness cost of the current position or can be defined as the average square error of the particles. Both of these properties are used for the entire solution searching process. In order to avoid the problems of the local optima, PSO adopted a concept of inertia weighting factor (IWF). This is a random function to solve the problem in PSO, which the optimal solution might jump into a local trap and cannot go out form the trap. So, this concept able to make the particles jump out from the local optima. This is also to increase

the convergence or accuracy rate (Laumanns, 2002). However, the operator IWF only for linear problems.

2.9 Multi-objective Particle Swarm Optimization (Coello, 2002)

2.9.1 Introduction

The strategy in MOPSO is to maximize the number of elements of the Pareto-front set found, minimize the distance of the Pareto-front produced by respect to global optimum and maximize the spread or distribution along the Pareto-front. The strategy is called as Archiving-method (AM). AM consists combination of three components. All of the elements in this technique are shown in Figure 2.21 and will be well-explained in the next section.

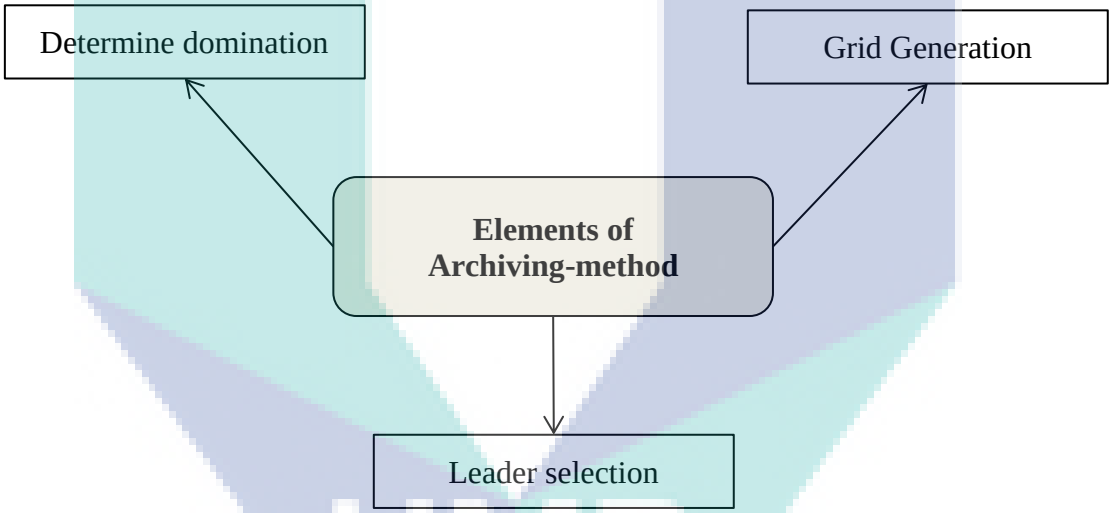

Figure 2.21 Three elements in Archiving-method (AM) approach.

2.9.2 Archiving-method

Archiving-method (AM) is one of the best techniques which adopted to PSO to make it deal with MO problems. In AM, the particles population characteristic is initialized at the beginning with "*Best Position*" criteria which denotes the best experiences or the best fitness value obtained by them. These values will be used to store POS generated previously.

In AM, a global repository is set up. This repository is the storage where the particles will deposit its movement experience after each iteration. Global attraction mechanism will be combined with the previous found POS that lead the convergence towards globally POS. The particles stored in the repository will be updated after each iteration. The best required number of solution (*i.e. 100 solutions will be stored in 100 of empty space in repository*) will be ranked, and particles that exceed from the repository space will be deleted. This top-ranked solution will be plotted and from this, Pareto-front can be generated.

2.9.3 Determine Domination

Unlike non-dominated sorting in NS method, to determine domination in AM is simpler. Determine Domination (DD) is an operator that will decide whether the particles in the current population is dominated or not. The Figure 2.22 below shows the procedure of DD.

Step 0: Preparation
Create population of a swarm, pop.

Step 1: Initialization
Set current particle domination as false, $pop(i).domination = 0$.

Step 2: Determine domination
Evaluate each particle with each other by following procedure.
If $pop(i) \leq pop(j)$, then $pop(j).Domination = true$,
If $pop(j) \leq pop(i)$, then $pop(i), Domination = true$

Figure 2.22 Determine domination (DD) procedure.

In this procedure, a swarm is required to be generated at first. In MOPSO, the swarm is evaluated at first based on the MO problem. This swarm then furthered to undergo this DD operation.

At beginning, the particles domination value will be set as '0'. The process then furthered to compare each particle to other particles based on the criteria in Step 2. After all particles has been evaluated based on their domination, they then will be separated and stored in a setup global repository (rep). In AM, all dominated solution will be deleted instantly. That make in this rep, only POS will be stored. This rep, which also

functions as an external memory will be put aside temporarily while the algorithms will operate and loops to generate new pop with new particles.

2.9.4 Grid Generation

The algorithm will then process to generate grid in the searching space. This strategy also known as relaxed forms of dominance or ε-dominance. This grid system is actually a strategy to preserve the distribution on Pareto-front. This geographically-based system grid will be generated after the algorithm calculates the fitness of each particle and mainly it operates to functional by filtering the POS in the external archive.

By ε-dominance, a user needs to define the desired number of grid division between two outermost found agents at left and right-hand side. This grid will form a range of small boxes and only one POS is retained in each box. The number of grids however need to be selected wisely because if the number chosen is too small, it will lead the low performance in terms of distribution of POS. However, the large value of grid division will affect the computation time, but the distribution will become well. Hence, the best optimum value is required to be defined.

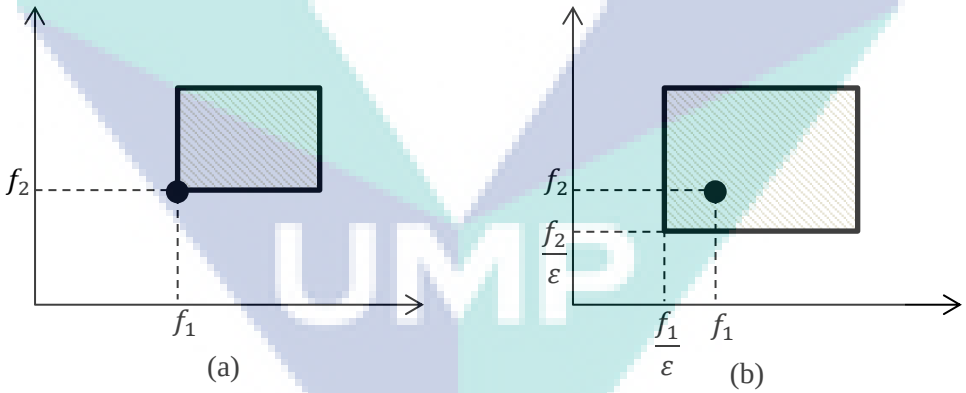

Figure 2.23 Normal dominance vs $\varepsilon - dominance$.

Figure 2.23 (a) shows that how certain area is dominated by certain solution using normal dominance technique. At the meanwhile, Figure 2.23 (b) illustrates the ε-dominance concept. The operation is to extend the current area by a value proportional to the parameter ε while ε is a constant value defined by user. The use of ε-dominance make sure that all the retained solutions are POS respect to all other found POS throughout the search. Important to denote that, when this bounding technique is

used, the size of final repository depends on the $\varepsilon - dominance$. Other than that, this technique also will make sure that it will select POS which better, if there are solution found better in the generated boundaries. The advantage is the algorithm could find the solution faster when this ε-dominance technique is adopted compared to normal dominance.

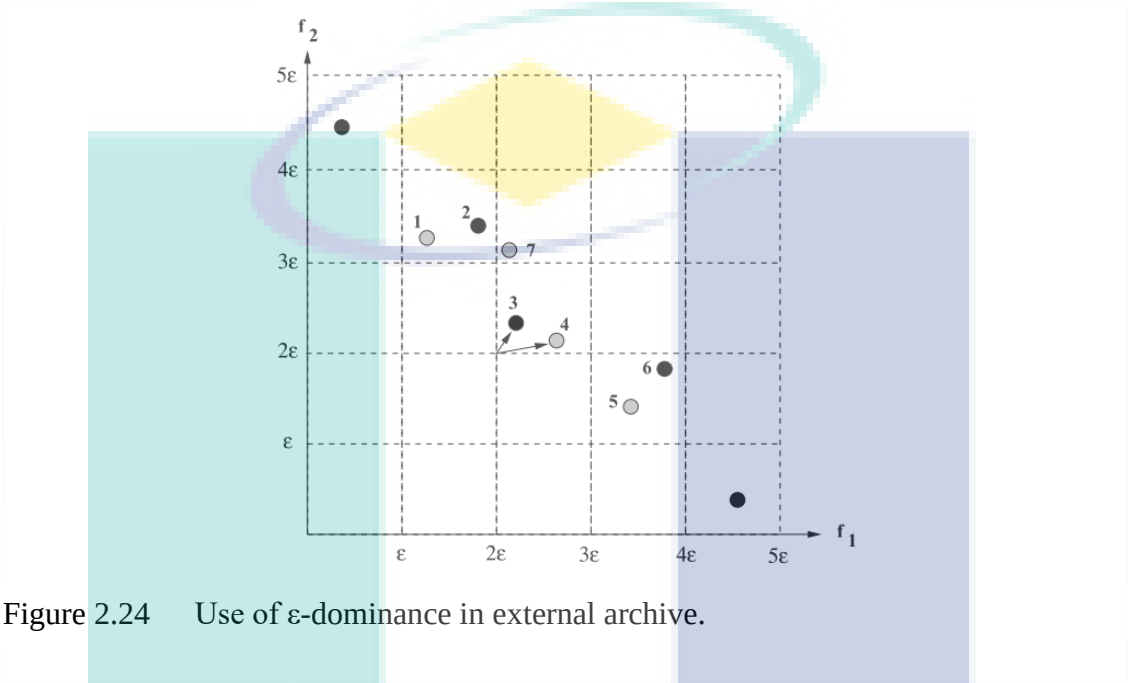

Figure 2.24 Use of ε-dominance in external archive.

Figure 2.24 show an illustration on how the operator ε-dominance operates in an external archive. To figure out, in the grid that consist of Solution 1 and 2, the operator will choose solution 1. This is due to Solution 1 is closer to grid vertices of ε and 3ε. For the meanwhile, in case of Solution 3 and 4, Solution 4 is preferred as it located closer to boundary of horizontal 2ε. For the grid consists of solution 5 and 6, it is clearly Solution 5 is more preferred as it is closer to the left-hand corner. For instance, Solution 7 is not accepted as solution 3 and 4 are located at the lower grid division compared to it. This brief how the operator filtering the POS in the external archive.

2.9.5 Leader Selection

It is important to emphasize that the leader selection is originally adopted from original PSO. However, the concept in PSO need to be changed as the solution in a MO problem consist of a compilation of good solution. For two objective problem, selection of leader is the key component of MOPSO.

The algorithm at the first place will evaluate the particles. From this evaluation, the POS found will be stored in the repository. Commonly, the approach is to consider all the POS in the repository are the leader and only an POS is selected. By this approach, a quality measure of the selected POS is really important in order to know how well it is.

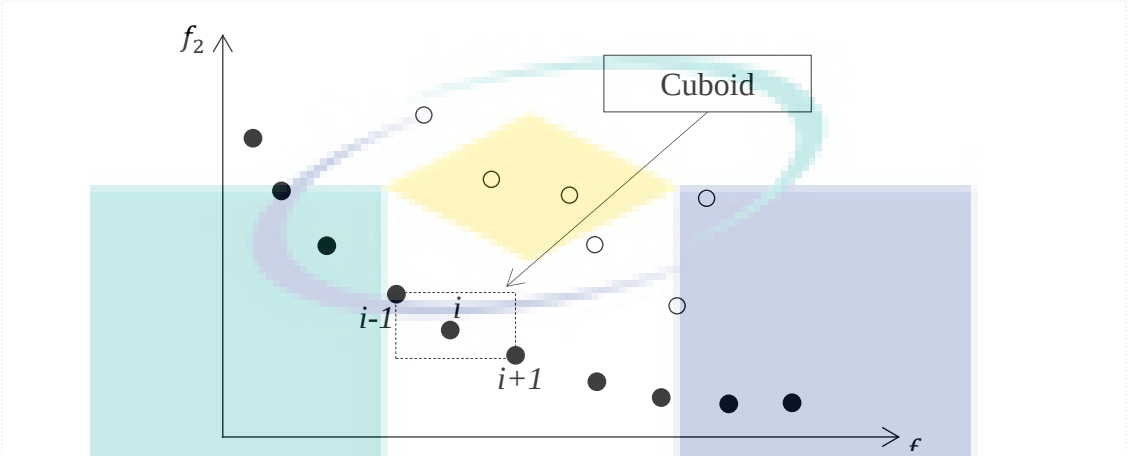

Figure 2.25 Nearest neighbour density estimator.

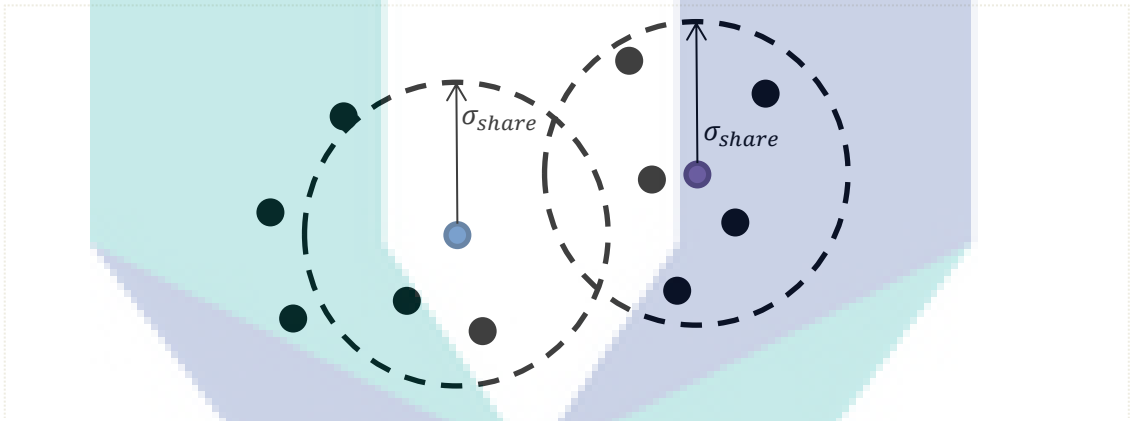

Figure 2.26 Kernel-density (KD) estimator

To do this, this quality measure can be related to density measures. By promoting diversity, the quality measurement of the POS can be known based on closeness of the particles within the swarm. Two types of most used density estimation are Nearest-neighbour-density estimator (NND) in Figure 2.25 and Kernel-density (KD) estimator in Figure 2.26. In NND, the techniques estimate how crowded are the closest neighbour of the POS in the objective function spaces. The strategy estimates the perimeter of the cuboid formed by using the nearest POS neighbours as vertices. This strategy is the same strategy of CD applied in NS-method. For instance, the fitness of a particle within a certain perimeter is degraded in proportion to the number and closeness to other particles that surround. when a particle sharing resources with others, So, in KD the strategy is defining a parameter called radius of neighbourhood, σ_{share}. It also indicates as niches.

Figure 2.28 graphically shows the overall procedure take place in AM. These procedures are taking place in one loop which will stop when the stopping criterion is met.

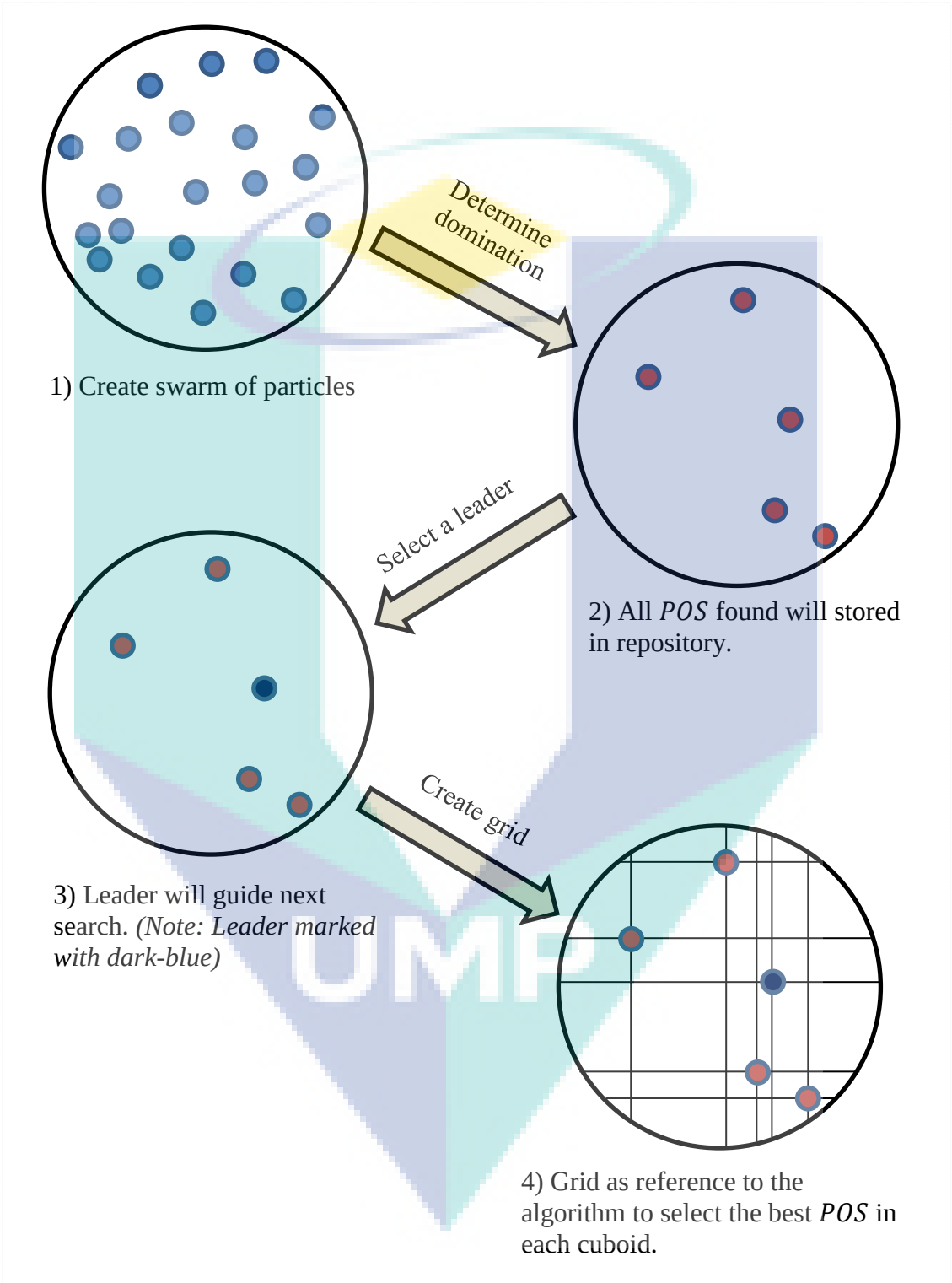

Figure 2.27 Illustration on elements in archiving-method over one generation.

2.9.6 General MOPSO

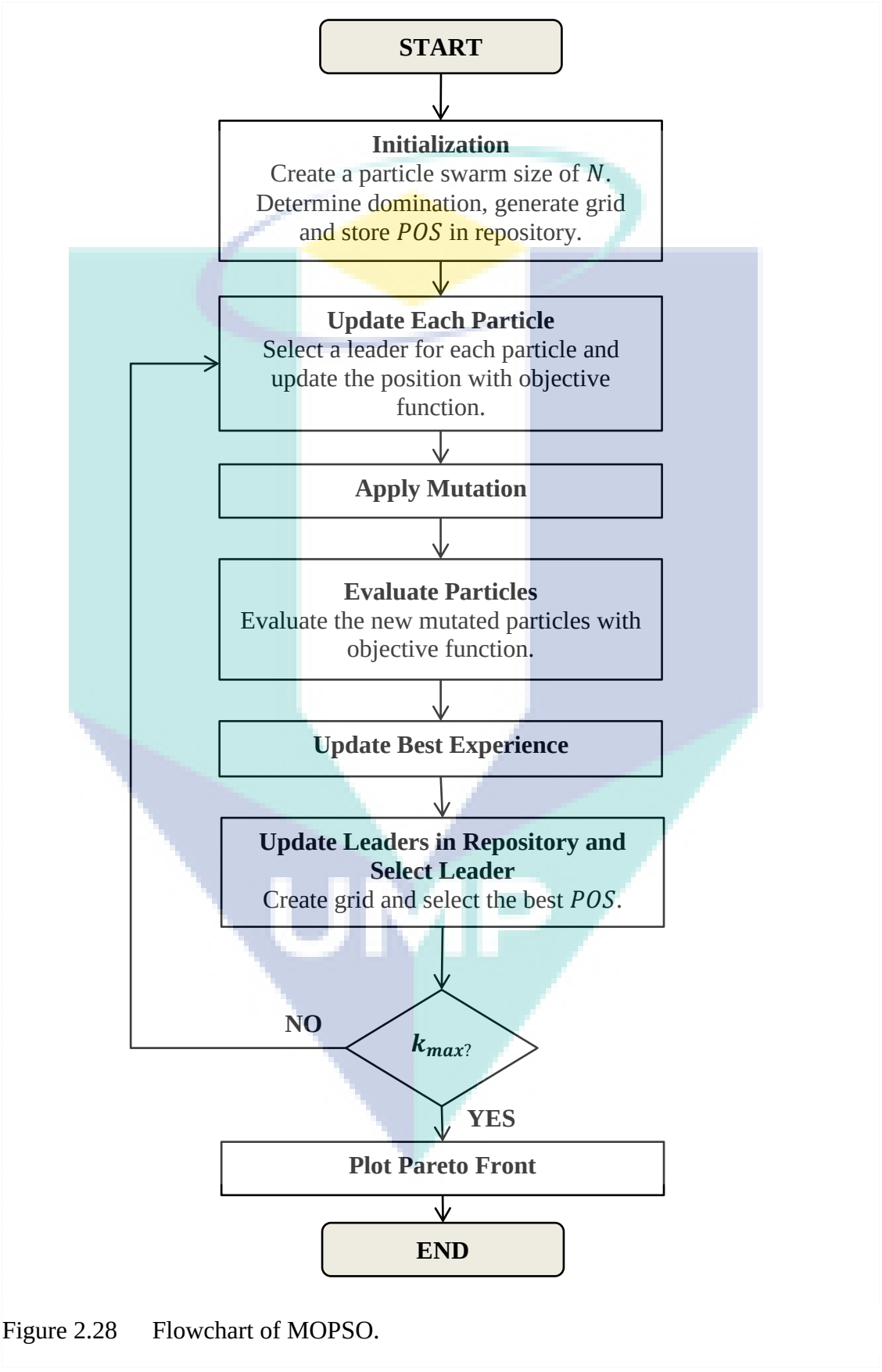

Figure 2.28 Flowchart of MOPSO.

2.10 Friedman Test (Pereira *et.al.*, 2015)

Friedman test is a non-parametric statistical test developed by Milton Friedman. It used to detect differences in treatments across multiple test attempts. In this test, each row will be ranked together, then the values of ranked will be considered according to column. Important to emphasize that the rows represent the block or individuals or matched sets of individuals while the column represent the various condition or treatments.

Friedman test operates with four simple steps. For a given data in form of a matrix with n rows, k columns and a single observation at the intersection of each block and treatment denoted as $\{e_{ij}\}_{n\times k}$. The ranks within the block will be calculated. If there are existence of tied values, the average of the ranks is assigned. This average value is without ties. The data is replaced with a new matrix of $\{w_{ij}\}_{n\times k}$ where the entry w_{ij} is the rank of e_{ij} within block i. Then, the procedure is furthered by calculating the average of matrix w using the following formulation in Equation 2.9.

$$\bar{w}_{\cdot j} = \frac{1}{n}\sum_{i=1}^{n} r_{ij} \qquad 2.9$$

After that, the equation of Friedman test statistic is applied. At this phase, value of F does not required to be adjusted for tied values in data by using Equation 2.10.

$$F = \frac{12n}{k(k+1)}\sum_{j=1}^{k}\left(\bar{w}_{\cdot j} - \frac{k+1}{2}\right)^2 \qquad 2.10$$

Final step, when n or k is large, for example for $n > 15$ or $k > 4$, the probability distribution of F can be approximated by that of a chi-squared distribution. In this case, $p - value$ is given by $P(\chi^2_{k-1} \geq Q)$. If n or k is small, the approximation to chi-square becomes poor and the p-value should be obtained from tables of F specially prepared for the Friedman test.

2.11 Wilcoxon Test (Cyprian, *et.al.*, 2015).

This test was named after Frank Wilcoxon, the man who proposed it. Similar to Friedman, Wilcoxon signed-rank test is also a non-parameter which used to compare to

related samples, matched samples, or repeated measurements on a single sample to assess whether their population mean ranks differ. Other than that, this test can be used to determine whether two independent samples were selected from populations having the same distribution.

Step 0: Preparation
Get two pairs of samples which consist of M number of individuals per pair. For each pair $i = 1, 2, 3, \dots, M$, let $m_{1,i}$ and $m_{2,i}$ as the notations. Assign H_o for "null hypothesis" in which the pairs that follows a symmetric distribution around zero and H_1 for which does not follow the symmetric distribution.

Step 1: Calculate difference and sign value of each pair
Calculate $|m_{2,i} - m_{1,i}|$ and $sgn(m_{2,i} - m_{1,i})$. Exclude pairs with $|m_{2,i} - m_{1,i}| = 0$. Assign M_r be a reduced sample size.

Step 2: Order and rank the remaining pairs
Order the remaining M_r pairs from smallest to largest absolute difference, $|m_{2,i} - m_{1,i}|$. Then, rank all the pairs using notation of R_i.

Step 3: Apply test statistic, W
Calculate the sum of the signed-ranks using formulation below.

$$W = \sum_{i=1}^{M_r} [sgn(m_{2,i} - m_{1,i}).R_i]$$

Step 4: Checking distribution criterion
Under null hypothesis, W follows a specific distribution with no simple expression. The expected value is 0 and a variance resulted by equation below. Using this equation, a reference table for the table is generated.

$$var_W = \frac{M_r(M_r + 1)(2M_r + 1)}{6}$$

Refer critical value of W from generated table. Perform two-sided test and reject H_o if $|W| > W_{critical,\, M_r}$.

Step 5: Calculate $z - score$
For $M_r \geq 10$, calculate $z - score$ using the formulation below.

$$z = \frac{W}{\sigma_W}$$

Where σ_W is as follow.

$$\sigma_W = \sqrt{\frac{M_r(M_r + 1)(2M_r + 1)}{6}}$$

Perform two-sided test and reject H_o if $|z| > z_{critical}$.

Figure 2.29 Pseudocode of Wilcoxon test.

The pseudocode of Wilcoxon test is shown in Figure 2.29. Wilcoxon test operates in five steps. For a briefing, let M be a sample size. To compare two samples, there will be $2M$. For pairs $i = 1, 2, 3, \dots M$, let $m_{1,i}$ and $m_{2,i}$ denote the comparison. The comparison resulted two types of differences between the pairs which are denoted as H_o and H_1. H_o is the difference between the pairs follow a symmetric distribution around zero while H_1 is vice-versa.

The next procedure is to calculate difference between individuals in each pair and their signums value using stated formulation in Step 1. At this phase, exclude all individuals with absolute difference of zero. The next Step 2 is to order and rank the remaining pairs. The operation will rank the pairs from smallest to largest value of the absolute difference. This rank will use notation of R_i.

In Step 3, each pair will be used to calculate Wilcoxon signed rank test by applying the formulation in the figure. As mentioned in the figure, under H_o, W follows a specific distribution with no simple expression. Their expected value is zero with variance resulted by the equation in Step 4. Using the equation, a specific reference table is generated. Then, the critical value of W is referred from the generated table. In this step, perform two-sided test and reject H_o if $|W| > W_{critical,\, M_r}$. The final step is to calculate $z - score$ and two-sided test is once more performed. This time, H_o will rejected if $|z| > z_{critical}$.

CHAPTER 3

Methodology

3.1 Introduction

This chapter explains about the whole methodology that has been used to design the algorithms. Solutions for the problems, the physical and computational domains of study will be clearly defined. In necessary, the governing equation used for the simulation, computational method and the procedure of solution will be also described in detail.

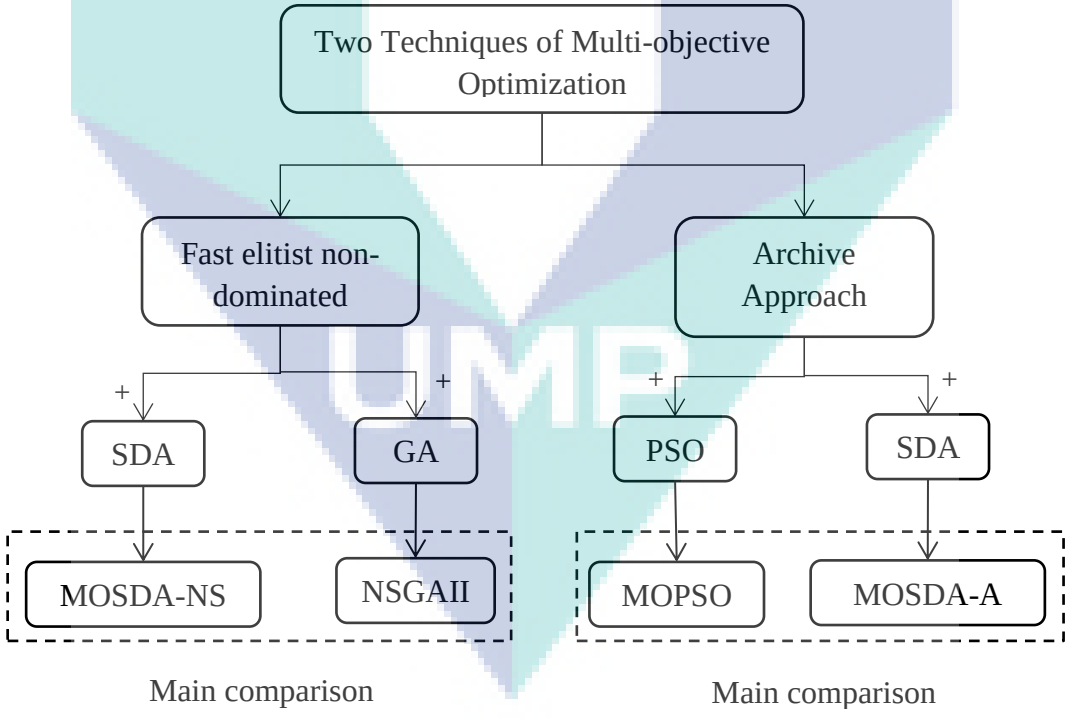

Figure 3.1 Origins of MOSDAs with its predecessors.

3.2 Project Flow

To understand how the research is done, the methodology is at the first place should be clarified. The methodology for developing this algorithm is divided into three phases as shown in Figure 3.2.

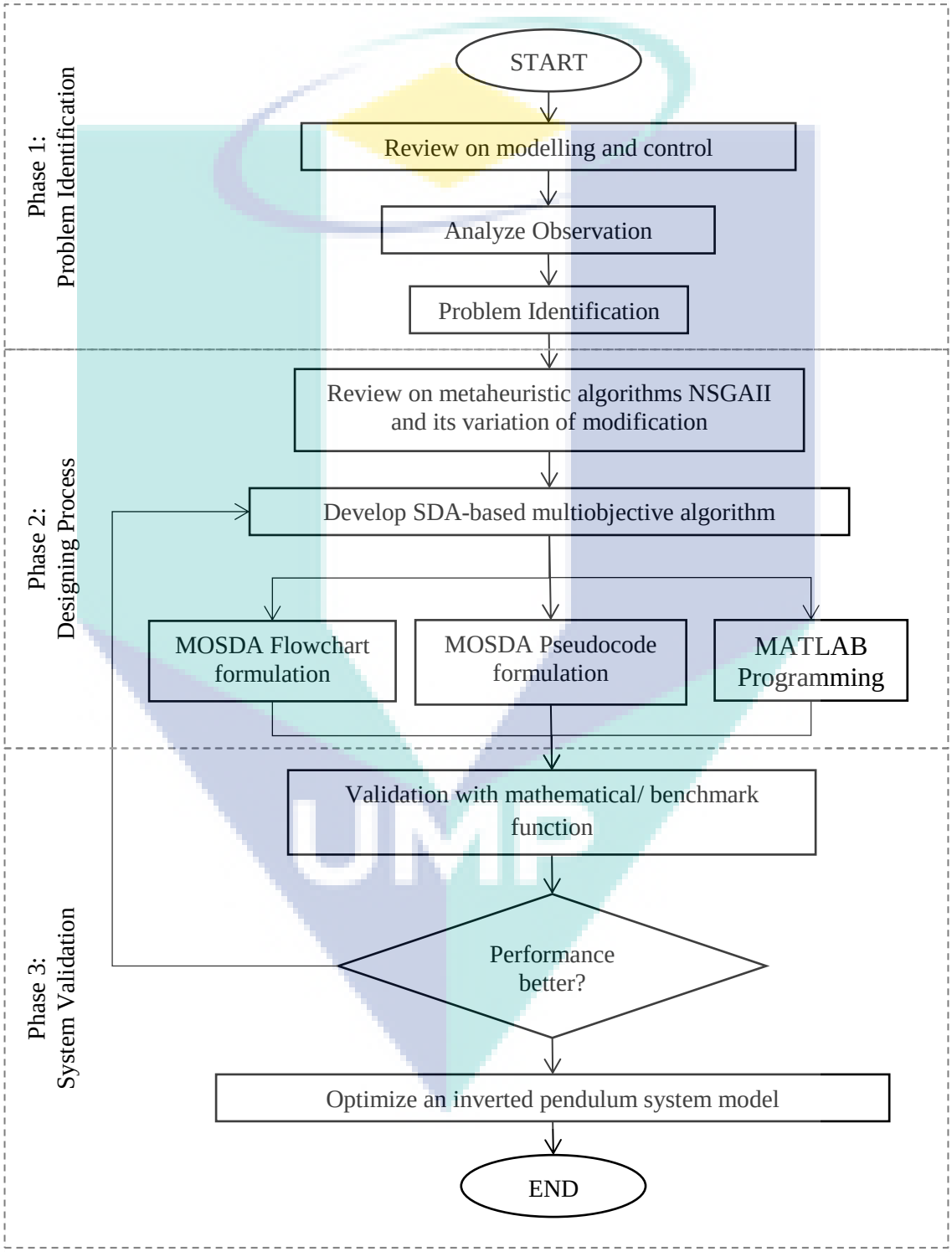

Figure 3.2 Project flow.

In the first phase, the research is to focus on the problem identification on current used algorithm in many fields. At first place, general view on modelling and control are discovered. They includes the algorithm which currently used to optimize parameters of robotics and optimize values in economics measurement (Bajd *et.al.*, 2010). Many papers and journals, are reviewed in order to identify the ineffectiveness, weakness and lacks in current optimization algorithms. To do this, the study reviews and analyses various strategies used in many algorithms.

The second phase will start to review on the metaheuristic algorithms. The review will more focus on both NSGAII and MOPSO and their applications to real world, the modification or upgrade done to the sequence of algorithm to provide more better solution set. The next step in this phase is to develop two SDA-based MO optimization algorithms. A Fast-elitist Non-dominated Sorting method (NS) and an Archiving-method (AM) will be used in the development. The development then will continue to specify the step to calculate the solution in the algorithm. This can be done by arranging the step involved using pseudocode. At the same time, the MATLAB is used to test each of the step whether it complies with the SDA or not. The developed algorithms will be then justified at the early stage with benchmark function. The result is gained by plotting the Pareto-front. The process will be furthered to validate the algorithms. After testing using benchmark functions the algorithm needs to be compared against other solution provided by other algorithms. If the performance is not satisfied the main objective, then the adjustment is needed to be done to the sequence of algorithm in order to achieve the objective, which is to provide a set of solution with better response. In the early phase, the algorithm will be tested with Schaffer MO problem. This is to ease the process in justify whether the algorithm could find the solution for this simplest problem. After the algorithm achieved to find its Pareto-front, then the algorithm will be furthered tested with other nine MO problems. From these 10 problems, six of them are more-than-two dimensional, which considered as the real challenge to MO algorithm to prove their ability. Hence, in next chapter there will be two subsections to discuss two categorial MO problems benchmarking test.

Third phase of the study is to apply the new developed MOSDA on a real-world application. In this thesis, MOSDA will be applied to an IP system. This testing will be done as simulation in MATLAB-Simulink software. It is very important to denote that

the simulation done will lead to deeper understanding of the algorithms purpose to the engineering field. This will add a valuable knowledge, which could lead better result at the end of MOSDA development.

3.3 Proposed-developed Multi-objective Spiral Dynamic Algorithms

3.3.1 Introduction

From the previous chapter, there are several methods from numerous papers which can be used to turn the SO-type algorithm into MO-type algorithm. The most successful method used by researchers are Fast-elitist Non-dominated Sorting (AM) and Archiving-method (AM) which used in NSGAII and MOPSO respectively.

As these methods are most strategic, they will be adopted into SDA in order to develop the MO SDA algorithm. In the next section, two new algorithms will be described well. The tree-chart in Figure 3.1 shows that the approach techniques from NSGAII and MOPSO, which then will be furthered adopted by SDA in order to make it as MO-type algorithm. The next section will describe the MOSDA-NS and MOSDA-A with detail.

3.3.2 MOSDA-NS: Non-dominated Sorting Spiral Dynamic Algorithm

3.3.2.1 Introduction

At the beginning, the algorithm will create a population of random agents. A population (pop) which consist of a number of (n_{pop}) searching agents will be generated. Important to emphasize that, this pop will be the main population that contain the best solution for each loop. There are also other two populations; Population 1 ($pop1$) and Population 2 ($pop2$) which will be the secondary and tertiary population to create maximum number of randomized parents' agent in the procedure. These population also will be useful to create interpopulation mutation and crossover children. Particularly, these population will contain agents with a few of specific characteristics or entities. To be clear, the searching agents in the generated population $pop, pop1$ and $pop2$ will be the same type of individuals which generated in NSGAII. These agents, will spread randomly onto the search spaces, according to the MO problem search ranges. The information of position of these new updated positions will be stored in the agent's entity.

Denote that, entities are a bunch of properties that an agent has. Table 3.1 shows the detail of the entities which every agent has.

Table 3.1 The entities for each designated agents in population of MOSDA-NS.

	Characteristic of an agent (m)
1	Position
2	Cost
3	Rank
4	Crowding Distance
5	Domination Set
6	Domination Count

Entity *"cost"* will present the cost value of particular position. As mentioned before, the agents which have a lower cost value are better because they are closer to the theoretical optimum value. By these cost values, the agents will be ranked in ascending order and their crowding distance (CD) further will be calculated. At the meanwhile, those entities called *"domination set"* an *"domination count"* will specify which fronts those agents belong to. These entities will be stored in every agent for every loop until the stopping criterion is achieved. The more loops set by the user, then more and better information gathered by the algorithms, and this will improve the result.

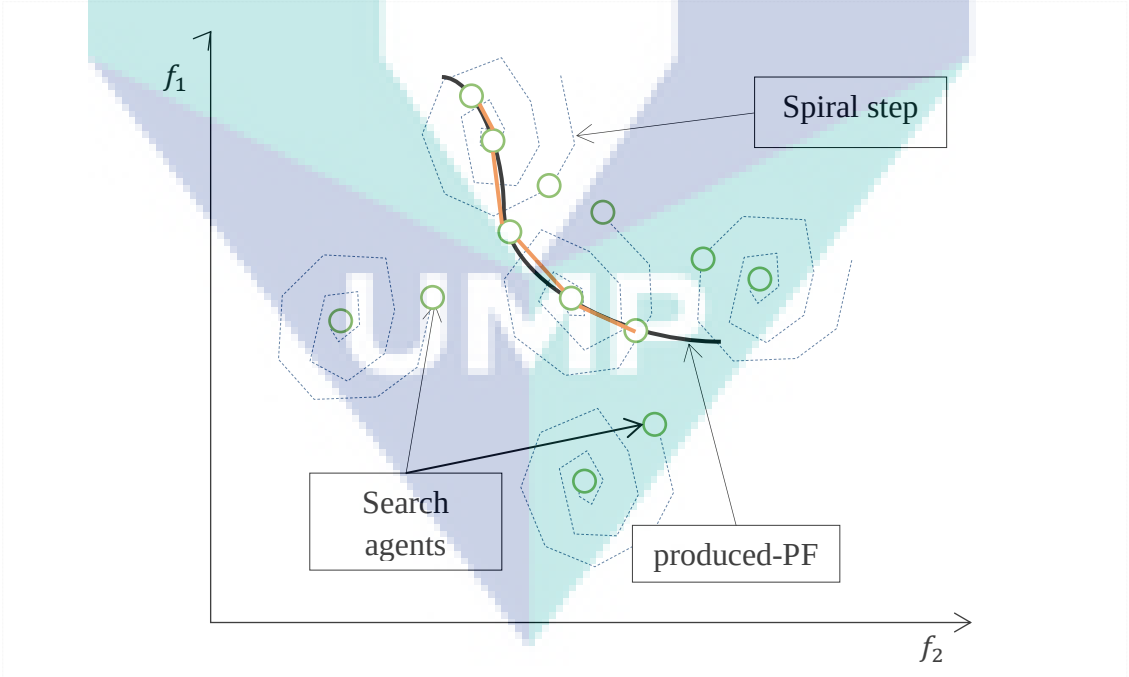

Figure 3.3 The agents spread and how they move in spiral step looking for POS.

Also, in this algorithm, a new approach by generating more than one population is introduced. As the consequence, Population 1 ($pop1$) will be generated. $pop1$ is generated based on the spiral model equation in Equation 2.6. This new position will be evaluated and its cost value will be stored into the agents' entity in $pop1$. Then, the algorithm continues to create new population or Population 2 ($pop2$). For this $pop2$, position of the agents in $pop1$ are taken as the reference. This $pop2$ position will be randomized by using Equation 3.1.

$$x_{2i} = (x_{1i} + 2\pi\, rand(\sin(X))(1 - \exp|x_{1i}|) \qquad 3.1$$

From the equation, x_{2i} is the new position of agents in $pop2$, x_{1i} is the current position of agents in $pop1$, where $i = 1, 2, 3, \dots, n_{pop}$ and X is the $b \times c$ matrix which contain of generated random values. This $pop2$ will be combined with pop and $pop1$ which further will undergo NS procedure. The products of SDA then will be transferred to next procedure of mutation and crossover.

Beyond introducing more than one population, there are also more randomization of the agents in MOSDA-NS compared to NSGAII. Even though SDA has a balanced exploration and exploitation strategies, but it is very easy to trapped in local optima, which is not good. This lead SDA operator in MOSDA-NS will have solution that has low performance in terms of its diversity. To have a well-diverse Pareto-front, global optimum is likely. Hence, the randomness strategies are really required. Despites, these random operators of mutation and crossover are inspired from NSGAII itself. However, not all members will be mutated and crossover. Only specified selected members are pass through this operator. In this MOSDA-NS, the specified members which will be passed through mutation and crossover are as stated in Table 3.2.

Table 3.2 Specified agents to mutate and crossover.

1st Mutation	First 5 members of pop and $pop1$
2nd Mutation	Last 5 members of pop and $pop1$
1st Crossover	First 5 members of pop and $pop1$
2nd Crossover	Last 5 members of pop and $pop1$

From the table, the algorithm run mutation and crossover for two times. Before this operator are operated, the population must be passed through NS procedure. This is to ensure the entire population are sorted according to the best fitness values. Other than

that, NS also used to obtain POS in desired amount. Hence, the current population that will be mutated and crossover are the population that has the best n_{pop} of POS found from the loop before.

3.3.2.2 Descriptive Main Loop of MOSDA-NS

From the Figure 3.4, the whole pseudocode for MOSDA-NS are shown. The algorithm will start at Step 0 with the user will need to define the number of search agents, m, radius, r and theta, θ of spiral $S_n(r, \theta)$, maximum number of iteration and number of function evaluation, k_{max} and nfe_{max} respectively and mutation rate, $mutrate$ and crossover rate. In Step 1, the algorithm then will create a population pop containing search agents that spread randomly in the feasible region. The ranges of this feasible region are specific based on the MO problem. Throughout the overall search, the agents only will randomly spread in this feasible range.

The algorithm will then recall the function of MO problem to calculate the cost value for each position. This cost then will be stored into the properties of each agent in pop. Then, the NS procedures will be applied to the pop. To recall, cost value will be used to determine which agents are POS. At this stage of initialization, the CD operator is also adopted. By using CD value, the algorithm will sort the population in ascending order. This procedure very important to determine the centre of the spiral, x^*. In Step 2, the first-ranked agent in current population will be chosen as the centre of spiral, x^*. This point will be a reference to other agents to move inwards in spiral step.

The algorithm then will update the position of the agents in Step 3. To do this, the spiral model equation is used. All agents will be moved in spiral towards centre, x^* a step ahead. These agents then will be evaluated to gain its cost value. The algorithm produced third population called $pop2$ in this step. This $pop2$ will be produced based on the random equation as stated in Equation 3.1. This approach is aimed to have more choices of POS. Important to mentioned that these $pop1$ and $pop2$ will be combined with the previous pop in order to find the best 50 POS.

Step 0: Preparation
Select a $m \geq 2$ number of search agents in a population pop, number of variables, n, parameters of spiral radius, r and angle of convergence, θ, maximum number of iterations, k_{max} and maximum number of function evaluation, nfe_{max}. Denote that, $n_{pop} = m$, and set mutation rate, $mutrate$.

Step 1: Initialization
Create a population, pop of agents with position $x_i(0) \in R^n$ which $i = 1, 2, 3 \ldots, m$. These agents will spread randomly in the feasible region.
The agents will be evaluated to determine their fitness.

$$Fitness\ cost = f_n\big(x_i(k+1)\big)$$

Apply NS procedures in Figure 2.16, 2.18 and 2.20. Sort population

Step 2: Define the centre of the spiral, x^*
The first-ranked agent in current population will be the centre, x^*. Later, all agents will move towards this agent in spiral step.

Step 3: Move agents a step ahead in spiral step
Update the position of every agents. The agents move in spiral step towards x^*. The new updated agents of $i = 1, 2, 3, \ldots, m$.

$$x_i(k+1) = S_n(r, \theta)x(k) - S_n(r, \theta) - I_n)x^*$$

Step 3: Randomize $pop1$ and create $pop2$
By using the agents in $pop1$, create new population 2, $pop2$ based on the random equation. Update new information of $pop2$.

$$x_{2i} = (x_{1i} + 2\pi\, rand(\sin(X))(1 - \exp|x_{1i}|)$$

Determine the fitness of the $x_i(k+1)$ of $i = 1, 2, 3, \ldots, m$.

$$Fitness\ cost = f_n\big(x_i(k+1)\big)$$

Step 4: Apply mutation procedure in Figure 2.7.
Mutate first 5 and last 5 agents in pop and $pop1$ as stated in Table 3.2.

Step 5: Apply crossover as in Figure 2.6.
Apply crossover to first 5 and last 5 agents in pop and $pop1$.

Step 6: Apply NS procedures as in Figure 2.16, 2.18 and 2.20.

Step 7: Sort population of Y

Step 8: Check termination criterion of k_{max} or nfe_{max}
Determine whether $k = k_{max}$ or $nfe = nfe_{max}$. If not, then repeat from Step 3.

Step 9: Generate Pareto-front.

Figure 3.4 Pseudocode of MOSDA-NS.

In Step 4 and Step 5, the mutation and crossover then will be applied to these populations. However, only pop and $pop1$ are involved and not all agents are mutated and crossover. These agents are simultaneously mutated and crossover for only their best 5 and worst 5 agents. This is aimed to reduce the computation cost of the algorithm. Once the operations were finished, NS procedures will take over. This time, the algorithm will find all possible POS in all populations generated from the previous $pop, pop1, pop2$. The new updated pop will be saved and sorted in ascending order. This sorting will leave only 50 bests POS. This pop will be looped for the next round of iteration, which also will be used as the reference to next loop of searching.

Step 8 will check the termination criterion, whether k_{max} or nfe_{max}, which one are achieved at first. However, in this thesis k_{max} is secondary compared to nfe_{max}. NFE is chosen as the termination criteria, which will provide a fair comparison among the tested algorithm in the thesis. After the algorithm detect $NFE = nfe_{max}$, then the process will be stopped and Pareto-front graph will be generated. For better visualisation, MOSDA-NS is also can be figured out in the flowchart in Figure 3.5.

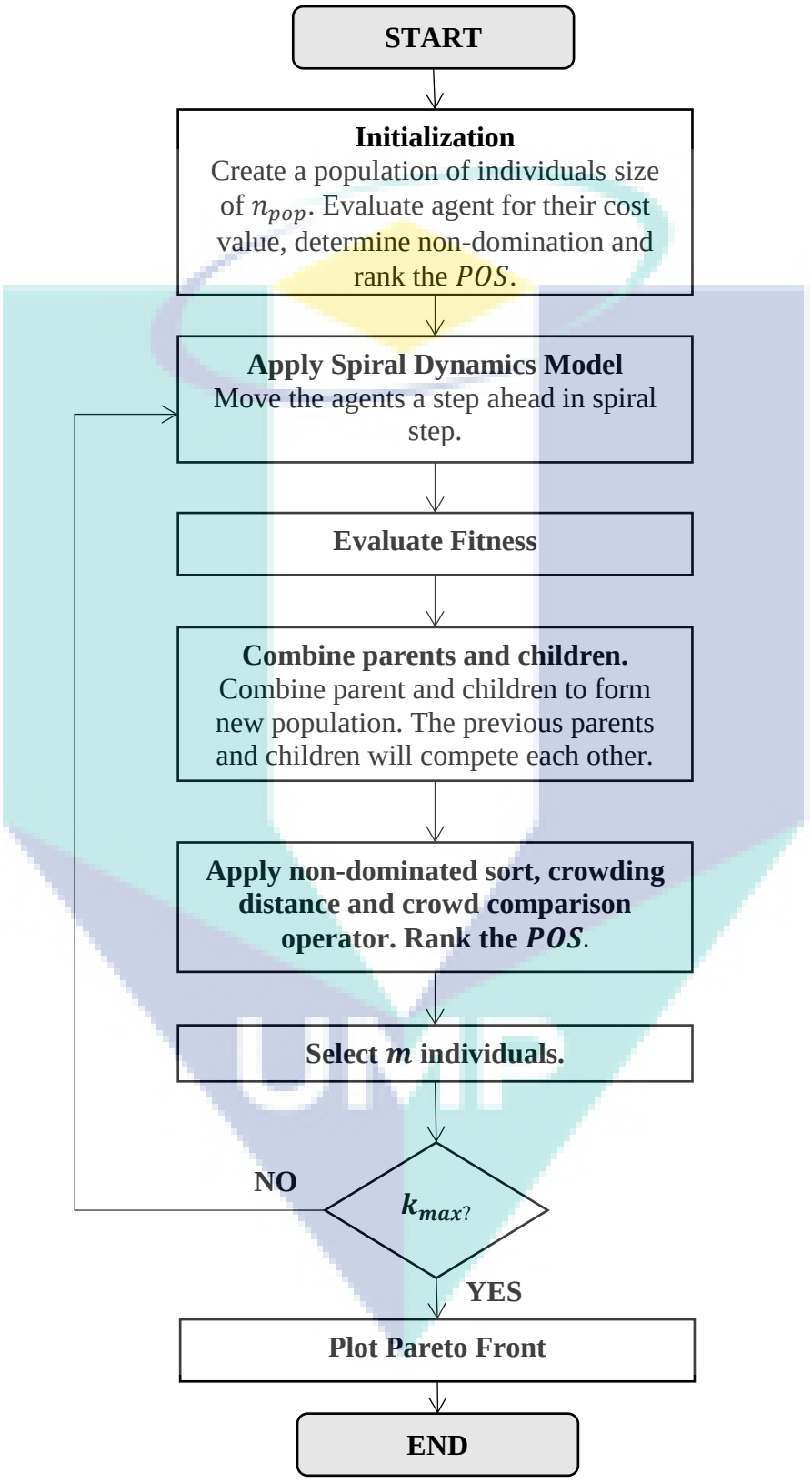

Figure 3.5 Flowchart for MOSDA-NS.

3.3.3 MOSDA-A: Archived-based Spiral Dynamic Algorithm

3.3.3.1 Introduction

This section introduces extension of SDA to solve MO problem by adopting Archive-method (AM). To find the solution for the problems, the Pareto-ranking scheme which previously explained by David E. Goldberg is adopted to the SDA. By that way, Pareto-dominance also applied to determine the flight direction of a particle, which will be explained well in next subsections. For some refreshment, this method is taken from MOPSO.

Equal with MOSDA-NS, a population of searching agents will be generated. Each agent will have specific characteristics as shown in Table 3.3. The best experience obtained by each agent will be saved in *"Global Best"* entity, while *"Domination"* is to save binary information; 0 for dominated and 1 for non-dominated. Grid Index and its Sub Grid Index are the information on the specific location of the agent, in term of x and y. This location will be used to create vertices of the grid.

Table 3.3 The entities for each designated agents in a population in MOSDA-A.

	An agent characteristic, m
1	Position
2	Cost
3	Global best
4	Domination
5	Grid Index
6	Sub Grid Index

There are eight steps in MOSDA-A. To briefly explain, the procedures that take place in MOSDA-A are quite similar to MOPSO. In addition, there are addition of elements to create more randomness in the searching agents is among the improvement. The elements are including mutation and crossover operators which taken from NSGAII. These operators chosen as their strategy is adaptable to various condition, universal and effective.

3.3.3.2 Descriptive Main Loop of MOSDA-A

Step 0: Preparation
Select the number of search agents, $m \geq 2$, number of variables, n, parameters of $S_n(r, \theta)$, radius, r and angle of convergence, θ, maximum number of iterations, k_{max} and maximum number of function evaluation, nfe_{max}. Denote that, $m = n_{pop}$ and set mutation rate, $mutrate$, number of grid division, $nGrid$ and crossover rate.

Step 1: Initialization
Create a population of agents with position $x_i(0) \in R^n$ which $i = 1, 2, 3 \ldots, m$. These agents will spread randomly in the feasible region.
The agents will be evaluated to determine their fitness.

$$Cost\ value\ = f_n\big(x_i(k+1)\big)$$

Determine domination (DD) of each agent using procedure in Figure 2.23.

$$POS = DD(Cost\ value)$$

Save POS found in global repository, rep.

$$rep(0) = POS(0)$$

Create grid and update the grid index for each POS.

Step 2: Define the centre of the spiral, x^*.
The first-ranked agent in rep will be set as the centre of the spiral, x^*.

Step 3: Apply spiral equation.
Update the position of every agents. The agents move in spiral step towards x^*. The new updated agents of $i = 1, 2, 3, \ldots, m$.

$$x_i(k+1) = S_n(r, \theta)x_i(k) - S_n(r, \theta) - I_n)x^*$$

Determine the fitness of the $x_i(k+1)$ of $i = 1, 2, 3, \ldots, m$.

$$Cost\ value\ = f_n\big(x_i(k+1)\big)$$

Step 4: Apply mutation and crossover.
Mutate and crossover first 2 agents and last 2 agents in rep. Calculate cost value.

Step 5: Update POS in rep.

$$rep(k+1) = POS(k) \cup POS(k+1)$$

Step 6: Update grid index and sub-grid index of each agents in pop.

Step 7: Check termination criterion of k_{max} or nfe_{max}.
Determine whether $k = k_{max}$ or $NFE = nfe_{max}$. If not, then repeat from Step 2.

Step 8: Generate Pareto-front.

Figure 3.6 Pseudocode of MOSDA-A

This explanation is based on the Figure 3.6. In Step 0, the algorithm needs some information. Most of the information are similar required in MOSDA-NS. The user needs to define m, r and θ of $S_n(r, \theta), k_{max}, nfe_{max}, mutrate$ and crossover rate. Other than

that, the user also needs to set desired number of grid division, $nGrid$. At the initialization of algorithm in Step 1, it will spread the searching agents in the feasible regions. These agents also will be randomly spread in this searching area at random manner between the MO problem search range. At this same phase, the agents will be evaluated. The cost value will be calculated using MO problem and this property will be stored into each agent entities. By the calculated cost value, the algorithm will determine domination. Each agent will be evaluated and those which found POS will be separated in external memory or global repository, rep. As stated before, rep will only store POS. The process will be continued to create grid. This grid also known as geographically-based system will be used to divide the current found POS area into several sections with boundaries. The system will create grid between the most negative and positive outermost of found position of POS.

The procedure continued to choose a centre of the spiral, x^* in Step 2. The first-ranked best agents stored in rep will be chosen to be x^*. At the next Step 3, the algorithm will update the current position of agents in pop. Similar to MOSDA-NS, at this step the spiral equation will be applied to move the agents a step ahead. This new position then furthered will be evaluated by calculating its cost value.

The next procedure in Step 4 is to apply crossover and mutation operation. This both operations are the same operations adopted in MOSDA-NS. By this operation, the new agents will be created and again to be evaluated to get its cost value. After this process done, the algorithm will update the POS in rep. At the meanwhile, the algorithm will update the grid and grid-index of each agents. This is stated in Step 6. Next, the process will be restarted from Step 2 as long as the termination criterion of k_{max} or nfe_{max} are not achieved. If the $k = k_{max}$ or $NFE = nfe_{max}$, then the algorithm will be terminated and Pareto-front graph will be generated.

To be clearer, flowchart of MOSDA-A also figured out in Figure 3.7.

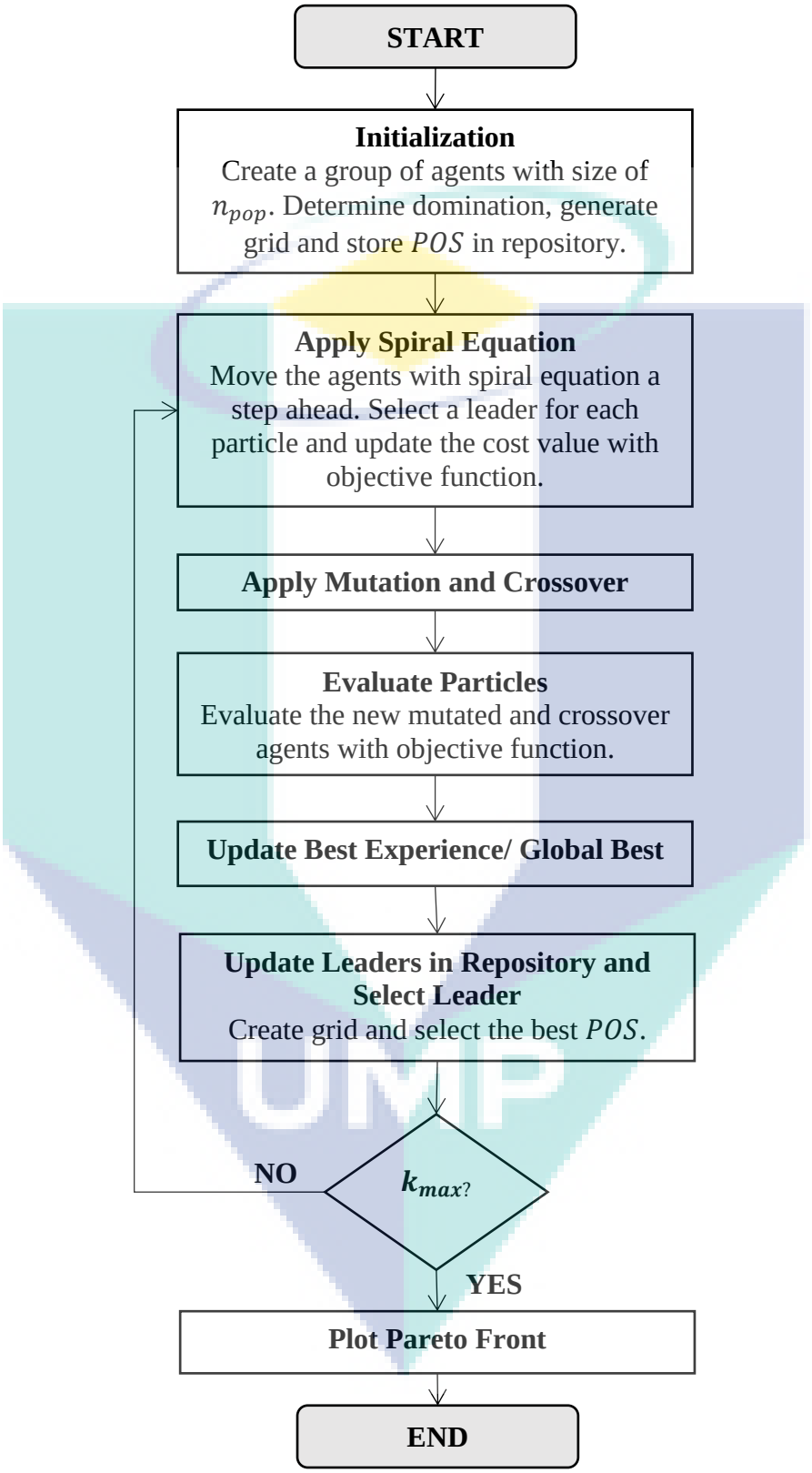

Figure 3.7 Flowchart of MOSDA-A.

3.4 Computer Simulation Setup

3.4.1 Hardware

The simulation was performed on a computer with Intel® Core™ i5-4440 CPU processor which running at 3.10GHz, 8Gb of RAM and a hard drive of 2Tb. The MOSDAs were coded in MATLAB and several parts of the codes were programmed in C++ language. The operating system of the PC used is Windows 8.1 Pro 64bit. This specification summarized in Table 3.4. To compare the results, a fair evaluation must be performed. Therefore, all of the tests are supposedly conducted on the same computer. About the development, both MOSDAs will be programmed in MATLAB and all experiment will be conducted on this specified hardware.

Table 3.4 The specification of computer used.

Processor	Intel® Core™ i5-4440 @ 3.10GHz
Hard drive	2 Tb
Installed RAM	8.00 Gb
System type	64-bit operating system, x64-based processor
Windows edition	Windows 8.1 Pro 64bit
MATLAB version	MATLAB Simulink R2013a

3.4.2 Parameters Comparison

The selection of the user-define parameters is important in determining the performance of these algorithms. So, the preparation parameters for MOPSO and NSGAII are set-up at their best (Deb *et.al.*, 2000; Coello *et.al.*, 2006). These parameter values are taken from their papers and journals. Table 3.5 and 3.6 shows the initialization parameters of these two algorithms.

On the other hand, the same parameters of NSGAII and MOPSO, which used in MOSDA-NS and MOSDA-A are also chosen from the best of both their predecessors (Coello *et.al.*, 2002; Zhao *et.al.*, 2009; Liu *et.al.*, 2013; Hoseini *et.al.*, 2016). For instance, it is really important to notify the comparison of these parameter. The number of search points in SDA, m is equivalent to number of populations, $nPop$ in MOPSO and NSGAII. It is set for 50. For more detailed comparison, the parameter comparison shown in Table 3.7.

Table 3.5 Best initialization parameter for MOPSO.

Initialization parameter	Value
Number of population, $nPop$	50
Number of repository, $nRep$	50
Number of grid per dimension, $nGrid$	7
Mutation rate, $mutrate$	0.1

Table 3.6 Best initialization parameter for NSGAII.

Initialization parameter	Value
Number of population, $nPop$	50
Crossover ratio, pc	0.8
Mutation ratio, pm	0.3

Table 3.7 Comparison of user-defined parameters.

Parameters	MOPSO	MOSDA-A	MOSDA-NS	NSGAII
Number of individuals in a population, $m = n_{pop}$	50	50	50	50
Size of repository, $nRep$	50	50	-	-
Number of grids per dimension, $nGrid$	7	7	-	-
Mutation rate, $mutrate$	0.1	0.1	-	-
Leader selection pressure, β	2	2	-	-
Crossover ratio, p_c	-	-	0.8	0.8
Mutation ratio, p_m	-	-	0.3	0.3
Inertia weight, ω	0.5	-	-	-
Inertia weight damping rate, ω_{damp}	0.99	-	-	-
Personal learning coefficient, c_1	1	-	-	-
Global learning coefficient, c_2	2	-	-	-
Radius, r	-	0.5	0.5	-
Theta, θ	-	$\frac{\pi}{4}$	$\frac{\pi}{4}$	-

3.4.3 Stopping Criterion

For stopping criterion, number of function evaluation (NFE) is used rather than number of iterations (NOI). This NFE is not same as NOI because problem function can be recalled for more than once in a single iteration. Furthermore, NFE is preferred rather than NOI because it could provide more information of the problem. Thus, if the NFE is set limited, then the amount of information that can be provided by the algorithm for a problem is also will be limited. This is better fair way on how to compare algorithms. So, for our study it was to be ran for the same number of maximum fitness evaluation. The

more times for the algorithm evaluating a problem, it has more chances to provide a better solution. NFE for each benchmark function is summarized in Table 3.8 (Umair *et.al.*, 2017; Engelbrecht *et.al.*, 2014).

Table 3.8 NFE for each benchmark function.

Notation	Benchmark Function	Maximum Dimension	NFE
f_1	Schaffer	2	30,000 times
f_2	Schaffer N2	2	30,000 times
f_3	Fonseca	2	100,000 times
f_4	Poloni	2	30,000 times
f_5	Kursawe	3	30,000 times
f_6	ZDT1	30	30,000 times
f_7	ZDT2	30	30,000 times
f_8	ZDT3	30	30,000 times
f_9	ZDT4	10	30,000 times
f_{10}	ZDT6	10	30,000 times

The method to set NFE is by observing the Pareto-front graphically and by looking at the number of POS found, which denotes its convergences toward the optimal Pareto-front. However, if the algorithm could not find POS for certain MO problems, therefore, the NFE then increased to a number in which all algorithm could find the POS at quickest. This NFE will be minimum NFE, which is 30,000 times. From the Table 3.8, all algorithm set with 30,000 times of NFEs except for Kursawe. Kursawe function which is only a three-dimensional MO problem however not a fast-converging problem. MOSDA-A requires 100,000 times of NFE to come out with a well Pareto-front. Therefore, all optimization for this problem will be set with this amount of NFE. This step is to ensure a fair comparison of performance for all of the algorithms.

It is important to stress that all of these parameters are set for all the test ran. They are required to be rightly chosen in order to make an equal comparison of all algorithm involved. The unsuitable values of parameters chosen would affect the analysis on the algorithm performance badly.

3.4.4 Performance Metric

Throughout this research, performance of the algorithms only goes to the aspect of metric measurement for the approximation sets in Pareto-front. These metrics are grouped in three, which are:

1) **Accuracy metrics:** Directly to measure the convergence of POS. To be clear, this metrics indicates how far is true-Pareto-front from produced- Pareto-front (Jiang *et.al.*, 2014).

2) **Diversity metrics:** Stressed that distribution and spread are two related entities closely. However, they are not too completely same. Spread referring the range of values covered by Pareto-front, some defined spread as the extent of the POS. Distribution directly refers to relative distance among the POS along the Pareto-front (Jiang *et.al.*, 2014).

3) **Cardinality metrics**: Refers to the number of POS found in a search. The best is a larger amount of found POS (Jiang *et.al.*, 2014).

In this experiment, the performance metrics: generational distance (GD), diversity metric delta (DMD), metric of spacing (MOS) and hyper-volume indicator (HV) are applied for quantitative and qualitative evaluation for the performance. Denote that, GD is to a type of accuracy metric while DMD and MOS are types from diversity metrics. At the meanwhile, HV is the performance measurement covering both accuracy and diversity. However, there are no evaluation based on the cardinality metrics because the number of required POS is initially set or defined by the user.

3.4.4.1 Use of Performance Metric

For instance, the Pareto-front or also known as attainment surfaces could provide a good visual qualitative assessment tool in terms of graphical. However, to provide a more formal assessment, these algorithms should be compared and analyse by using statistical testing procedure. This is very important in order to define whether the algorithms are improved or comparable.

In this research, the inspection by observing the visual in Pareto-front plot will be done. Result will be reported in two-dimensional plots and the test will be stated in the tables, plus the statistically significant of any comparison will be also described. GD, DMD, MOS, HV and TOC are all useful as they allow another form of comparison between two or more algorithms.

3.4.4.2 Generational Distance (GD)

For explanation, Figure 3.7 shows the illustration for GD. GD is a parameter to measure the convergence between the true-Pareto-front and produced-Pareto-front. GD is actually the average of Euclidian distance between these two both of true- and produced-Pareto-front. This measurement proposed by Veldhuizen (Jiang *et.al.*, 2014). In term of mathematical formulation, the derivation of GD is shown in Equation 3.2.

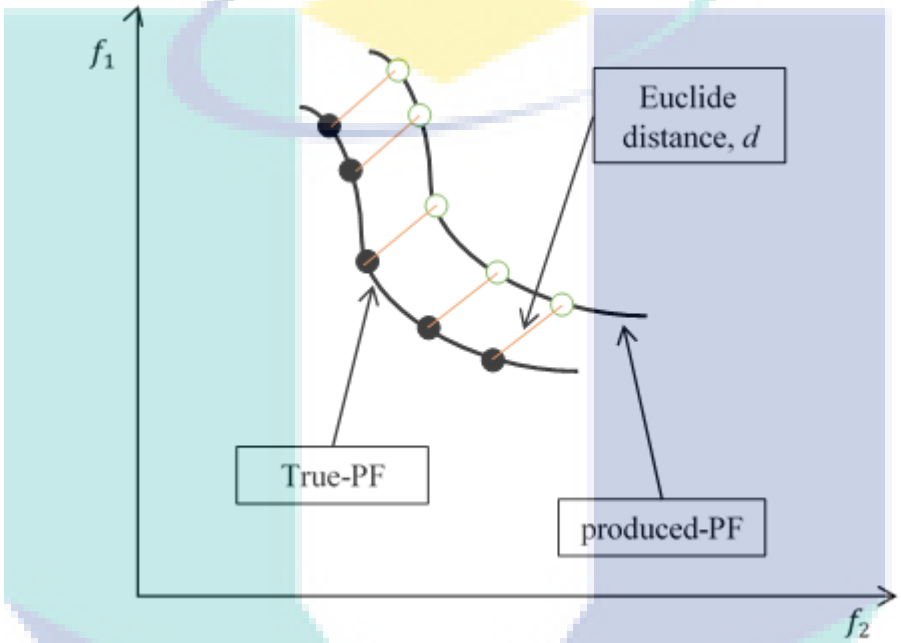

Figure 3.8 Illustration of GD.

$$GD = \frac{\left(\sum_{i=1}^{|Q|} d_i^p\right)^{\frac{1}{p}}}{|Q|} \quad 3.2$$

$$d_i^p = \min_{k \in |P^*|} \sqrt{\sum_{n=1}^{N} \left(f_n^i - f_n^{*i}\right)^2} \quad 3.3$$

From the Equation 3.2 and 3.3, Q is the produced Pareto-front, p is the number of comparing Pareto-front sets, d_i^p is Euclidian distance, i is specified number of produced NDS, P^* is true-Pareto-front, f_n^{*i} is n^{th} objective function of k^{th} member in P^*. For $p = 2$, the Euclidian distance, d_i between the NDS $i = Q$ and the nearest P^*, in the searching

space is in Equation 3.3. By this definition, the algorithm which has smaller GD value produces Pareto-front solution with better accuracy.

3.4.4.3 Diversity Metric Delta

In MO optimization, rather than achieving the accurate Pareto-front, the other main concern is to have a set of POS that spans the entire Pareto-front. The diversity metric delta (DMD) is the tool to measure the extent of spread achieved along the Pareto-front (Hien, N. T., *et.al.*, 2006; Jiang *et.al.*, 2014). Mathematical formulation for DMD is defined in Equation 3.4.

$$DMD = \frac{\sum_{n=1}^{N} d_n^e + \sum_{i=1}^{Q} |d_i - \bar{d}|}{\sum_{n=1}^{N} d_n^e + Q\,\bar{d}} \qquad 3.4$$

From the equation, Q is the produced Pareto-front, d_i is Euclidian distance between two consecutives POS , $\bar{d}$ is the mean for all distance and d_n^e is the distance between extreme solution of both true- and produced-Pareto-front respect to n^{th} objective. The Figure 3.9 shows the illustration of DMD. In the figure, the extreme solution is the two outermost POS found. By the above mathematical definition, the spread of Pareto-front will be better if DMD test computed a small value. Thus, the smaller the value, the better the spread.

3.4.4.4 Metric of Spacing

Metric of spacing (MOS) will calculate relative distance measure between consecutive solution in the gained POS (Jiang *et.al.*, 2014). To be clear, MOS is actually representing the measurement of standard deviation of different d_i. Figure 3.8 illustrates the MOS graphically.

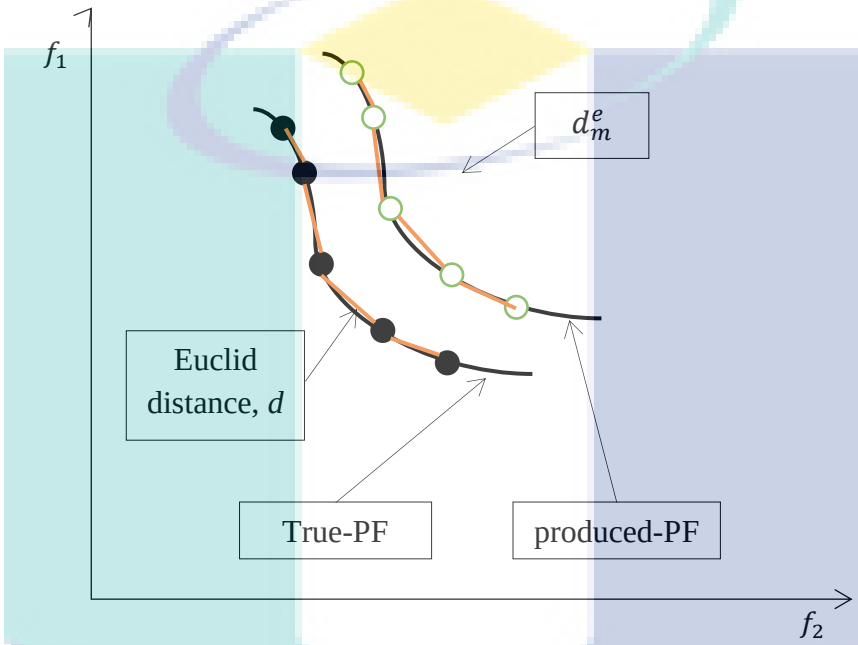

Figure 3.9 DMD and MOS illustration.

The mathematical formulation of MOS is as in Equation 3.5.

$$MOS = \sqrt{\frac{1}{Q}\sum_{i=1}^{Q}(d_i - \bar{d})^2} \quad 3.5$$

Where d_i is defined as in Equation 3.6.

$$d_i = \min_{k \in Q \wedge k \neq i} \left\{ \sum_{m=1}^{M} |f_m^i - f_m^k| \right\} \quad 3.6$$

and the mean value of the distance measure, $\bar{d}$ is stated in Equation 3.7.

$$\bar{d} = \frac{\sum_{i=1}^{Q} d_i}{Q} \quad 3.7$$

The $\bar{d}$ is the minimum value of sum of the absolute difference in fitness value between the i^{th} solution with any of other solution in the produced POS. Bold that, this $\bar{d}$ is not same with the minimum value of the Euclidian distance between two POS.

As it is a standard deviation, the smaller value of MOS indicate that the POS solution are uniformly better distributed along the Pareto-front.

3.4.4.5 Hypervolume Indicator

One of the test applied to these algorithm is hyper-volume indicator (HV) (Bradstreet, 2011; Cao *et.al.*, 2015). This tool in a combined sense computes a qualitative measure of both accuracy and diversity. Last but not least, this indicator useful to obtain a better overall picture of algorithm performance alongside with GD and DMD.

HV, also known as the hyper-area or $Lebesgue$-measure, which covering the entire objective space or feasible region that contain the approximate sets. The hyper-volume metric is also in other way can be defined as a measurement of the volume of the enclosed space between a reference point defined by the user and the weakly dominated portion of the objective space (Hien, N. T., *et. al.*, 2006; Riquelme *et. al.*, 2015). The HV operation is started by selecting a reference point, G and the solution i. Solution i will be the vertices of the hypercube. At the meanwhile, reference point, G is selected by constructing a vector of worst fitness value. For the mathematical formulation, HV can be defined as in Equation 3.8.

$$HV = volume\left(\bigcup_{i=1}^{Q} v_i\right) \tag{3.8}$$

The example is shown in Figure 3.8, two hyper-volumes A and B are generated from two different Pareto-fronts, which both are true and produced-Pareto-front respectively. In this example, HV B is better than A since B's HV is larger (Angus, 2008). The larger HV indicates that the Pareto-front is closer to 0 and better in its accuracy and diversity.

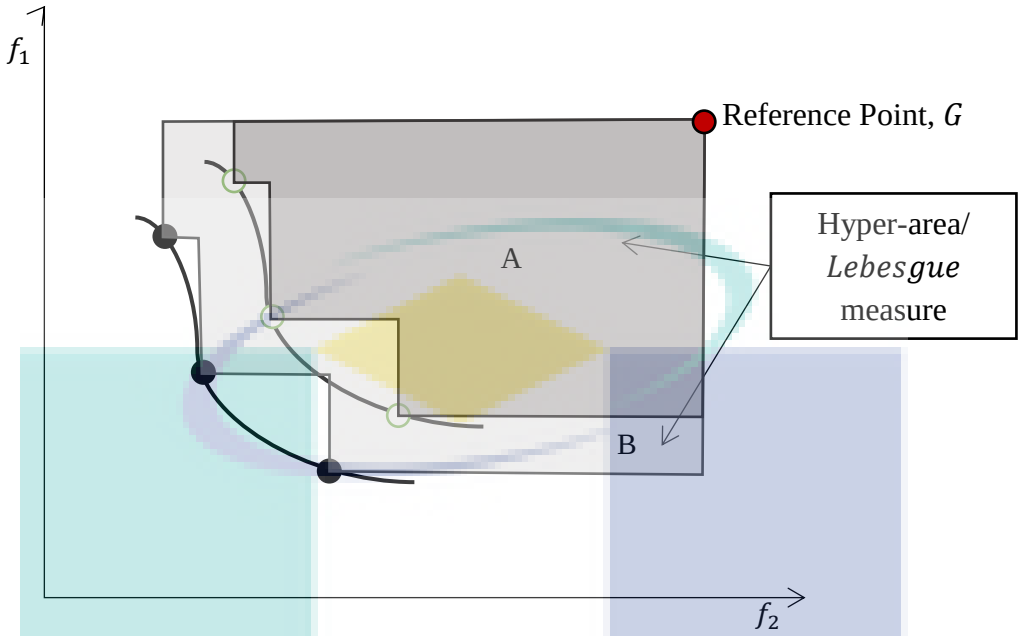

Figure 3.10 Illustration of hypervolume operators.

The indicator will provide the best HV if the value is getting larger. In the setup program, HV will compare between the true-Pareto-front and produced-Pareto-front. Therefore, the value provides by HV will be in ratio of 1 respect to true-Pareto-front. That means, the maximum value indicated by HV will be 1 while the worst value will be 0.

3.4.4.6 Time of Computation

In this research, time to computation (TOC) is also take into account. Time is crucial in many ways, including in this optimization area. The time is measured from the first iteration to any set maximum iteration. The techniques in this research is to measure the whole of iterations for each of run. Important to denote that, the total number of tests for each MO problem are 25 runs. From the data obtained from the runs, the average then calculated. Denote that the time measured is not for the specific NOI, by means actually it is the period measured to generate Pareto-front for specified NFE. Even though the number of NFE is same for all algorithms, the time taken for each of them might be different. This caused by the different or complex procedures need to be ran by the algorithms. Some complex procedures that involved might extend the time. Therefore, the more complex the algorithms, the longer times needed.

3.4.5 Benchmark Function

3.4.5.1 Introduction

For the benchmark study, the standard MOP are f_1 to f_5. These functions are the low-dimension MO problems. At the meanwhile, high-dimension MOP used to test all MO algorithm are ZDTs. ZDTs is a broad and well-known set of MO problems used to benchmarking the performance of MO algorithm in category PoM (Chase, *et.al.*, 2009). Furthermore, each of ZDTs' MO problems consist of specific character that represent the real-world application, in which might be face a harsh challenge in convergence towards the Pareto-front.

3.4.5.2 Standard Function

1) $\boldsymbol{f_1}$: Schaffer

Schaffer is a one-dimensional benchmark function. It is very simple test function for MOA (Schaffer, 1985). Even though this function is a simple problem, it is always has been applied to test the new developed algorithm and it is the best option to investigate the diversity of the population along the Pareto-front (Zitzler *et.al.*, 1998). The definition of objective functions is shown in Equation 3.9 and 3.10.

$$f_1(x) = \theta^2 \tag{3.9}$$

$$f_2(x) = (\theta - 2)^2 \tag{3.10}$$

It has search ranges between $5 \leq x \leq 5$.

From theoretical solution, it has a Pareto-front that continuous. From theory, the result forms a continuous curve which range from $0.0 < x < 4.0$ and $0.0 < y < 4.0$.

2) $\boldsymbol{f_2}$: Schaffer N2

Schaffer N2 is a two-dimensional MO problem. It has a discontinuous front. The definition of objective functions is shown in Equation 3.11 and 3.12.

$$f_1(x) = \begin{cases} -x, if\ x \leq 1 \\ x-2, if\ 1 < x \leq 3 \\ 4-x, if\ 3 < x \leq 4 \\ x-4, if\ x > 4 \end{cases} \quad 3.11$$

$$f_2(x) = (x-5)^2 \quad 3.12$$

It has search range between $-5 \leq x \leq 5$.

3) $\boldsymbol{f_3}$: Fonseca

Fonseca is a two-dimensional functions (Fonseca *et. al.*, 1995). The definition of objective functions is shown Equation 3.13 and 3.14.

$$f_1(x) = 1 - exp\left(-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right) \quad 3.13$$

$$f_2(x) = 1 - exp\left(-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right) \quad 3.14$$

It has the search ranges between $-4 \leq x \leq 4$, the $1 \leq i \leq n$.

From its problem theoretical solution, it has a Pareto-front that continuous. Theoretically the function will provide a convex-curve pareto front that ranged between $0.0 < x < 1.0$ and $0.0 < y < 1.0$.

4) $\boldsymbol{f_4}$: Poloni

Poloni is a two-dimension function problem. The definition of objective functions is shown in Equation 3.15 and 3.16.

$$f_1(x,y) = [1 + (A_1 - B_1(x,y))^2 + (A_2 - B_2(x,y))^2] \quad 3.15$$

$$f_2(x,y) = (x-5)^2 \quad 3.16$$

Where A_1, A_2, B_1 and B_2 are defined and listed in Equations 3.17.

$$A_1 = 0.5sin(1) - 2\cos(1) + \sin(2) - 1.5\cos(2) \quad 3.17$$

$$A_2 = 1.5sin(1) - \cos(1) + 2\sin(2) - 0.5\cos(2)$$

$$B_1(x,y) = 0.5\sin(x) - 2\cos(x) + \sin(y) - 1.5\cos(y)$$

$$B_2(x,y) = 1.5\sin(x) - \cos(x) + 2\sin(y) - 0.5\cos(y)$$

The search range is between $-\pi \leq x, y \leq \pi$. From this problem theoretical solution, it has a Pareto-front that discontinuous.

5) f_5: Kursawe

Kursawe is a three-dimensional problem. Definition is shown in Equation 3.18 and 3.19.

$$f_1(x) = \sum_{i=1}^{2}\left[-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right] \quad 3.18$$

$$f_2(x) = \sum_{i=1}^{3}[|x_i|^{0.8} + 5\sin(x_i^3)] \quad 3.19$$

The search ranges are between $-5 \leq x \leq 5$ and $1 \leq i \leq 3$. From this problem theoretical solution, this MO problem has a Pareto-front that discontinuous.

3.4.5.3 ZDT-based Function

All of the functions in ZDT have two objectives. This feature is the usually used in engineering and economics, which also most common application of Pareto-efficiency.

6) f_6: ZDT1

The ZDT1 function has a convex Pareto-front. The definition of objective functions is shown in 3.20 to 3.23.

$$f_1(x) = x_1 \quad 3.20$$

$$f_2(x) = g(x)\,h\big(f_1(x), g(x)\big) \quad 3.21$$

$$g(x) = 1 + \frac{9}{29}\sum_{i=2}^{30} x_i \quad 3.22$$

$$h\left(f_1(x), g(x)\right) = 1 - \sqrt{\frac{f_1(x)}{g(x)}} \quad 3.23$$

The search range is between $-0 \leq x \leq 30$.

7) f_7: ZDT2

The ZDT2 function has a non-convex Pareto-front. In this ZDT2 function, thirty design variables were chosen ($n = 30$). Each design variable ranged in value from 0 to 1. The definition is as in Equation 3.24 to 3.27.

$$f_1(x) = x_1 \quad 3.24$$

$$f_2(x) = g(x)\, h\left(f_1(x), g(x)\right) \quad 3.25$$

$$g(x) = 1 + \frac{9}{29}\sum_{i=2}^{30} x_i \quad 3.26$$

$$h\left(f_1(x), g(x)\right) = 1 - \left(\frac{f_1(x)}{g(x)}\right)^2 \quad 3.27$$

It has the search range between $-0 \leq x \leq 30$.

8) f_8: ZDT3

The ZDT3 function adds a discreteness feature to the front. Its Pareto-front consists of several non-contiguous convex parts. The introduction of a sine function in this objective function causes discontinuities in the Pareto-front, but not in the parameter space. In this ZDT3 function, thirty design variables were chosen ($n = 30$). Each design variable ranged in value from $0\ to\ 1$. The definition of objective functions is in Equation 3.28 to 3.31.

$$f_1(x) = x_1 \quad 3.28$$

$$f_2(x) = g(x)\, h\left(f_1(x), g(x)\right) \quad 3.29$$

$$g(x) = 1 + \frac{9}{29}\sum_{i=2}^{30} x_i \quad 3.30$$

$$h\big(f_1(x), g(x)\big) = 1 - \sqrt{\frac{f_1(x)}{g(x)}} - \left(\frac{f_1(x)}{g(x)}\right) sin(10\,\pi f_1(x)) \quad 3.31$$

This MOP has the search range between $-0 \leq x \leq 30$.

9) $\boldsymbol{f_9}$: ZDT4

The ZDT4 function has 21 local Pareto-fronts and therefore, it is highly multi-modal. In this ZDT4 function, the maximum ten design variables are being chosen ($n = 10$). The design variable ranges are from -5 to 5 for the last nine design variables and 0 to 1 for x_1 . The definition of objective functions is in Equation 3.28 and 3.29.

$$f_1(x) = x_1 \quad 3.32$$

$$f_2(x) = g(x)\, h\big(f_1(x), g(x)\big) \quad 3.33$$

$$g(x) = 91 + \sum_{i=2}^{10} (x_i^2 - 10cos(4\pi x_i)) \quad 3.34$$

$$h\big(f_1(x), g(x)\big) = 1 - \sqrt{\frac{f_1(x)}{g(x)}} \quad 3.35$$

This MOP has the search range between $-0 \leq x \leq 10$.

10) $\boldsymbol{f_{10}}$: ZDT6

The ZDT6 function has a non-uniform search space, whereas the POS are non-uniformly distributed along the global Pareto-front, and also the density of the solutions is lowest near the Pareto-front and highest away from the front. In this ZDT6 function, ten design variables were chosen ($n = 10$). The design variable ranges are from 0 to 1. The global Pareto-optimal front appears when $g = 1.0$. The definition of objective functions is in Equation 3.36 to 3.39.

$$f_1(x) = x_1 \tag{3.36}$$

$$f_2(x) = g(x)\, h\big(f_1(x), g(x)\big) \tag{3.37}$$

$$g(x) = 1 + 9\left[\frac{\sum_{i=2}^{10} x_i}{9}\right]^{0.25} \tag{3.38}$$

$$h\big(f_1(x), g(x)\big) = 1 - \left(\frac{f_1(x)}{g(x)}\right)^2 \tag{3.39}$$

The search range is between $-0 \leq x \leq 10$.

3.5 Validation of Proposed Algorithm

This chapter describes comprehensively all 10 MO problems used to validate the performance of proposed MOSDA-NS and MOSDA-A. These benchmark functions are selected because of its significant contribution of past studies in this area. They are coded in MATLAB alongside with proposed algorithms. For a fair comparison, NSGAII and MOPSO also ran again with these MO problems to obtain the results from same specification of hardware. To make the comparison, these algorithms executed the MO problems in 25 independent runs. Additionally, the performance test will be furthered for a simulation of inverted pendulum (IP) system. All algorithms will be used to perform the optimization task for the PD-controller of this IP system. The results will be described in detail in the next sections.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

Several experiments that conducted to evaluate the performance of the proposed algorithms are presented in this chapter. The experiment will compare the performance of proposed algorithms with the original version of NSGAII and MOPSO. At first, it is important to highlight that NSGAII is the predecessor of MOSDA-NS while MOPSO is the predecessor of MOSDA-A. The comparison between new algorithms and their predecessor will be focussed. However, the overall comparison involving all four algorithms is also done. This comparison will be presented well.

The experiments carried out involving all algorithms, 1) MOSDA-NS, 2) MOSDA-A, 3) MOPSO and 4) NSGAII. To clarify, MOPSO and NSGAII are also run again in order to make a fair comparison by using the same hardware specification. All of the algorithms test depends on the characteristic of the MO problem. Some MO problem are simple and easy to solve while some are complex structured. The mathematical formulation of the MO problem justify the complexity of themselves, including their number of dimensions. To stress is, all function is tested with their maximum dimensions. To organized the analysis, these MO problem are divided into two groups; 1) Two-dimension MO problem and 2) More-than-three dimensional MO problem.

In this section, all of the experiments-gained data will be described well. The simplest way to analyse an optimization solution is by looking at the generated Pareto-front. The Pareto-front or also known as attainment surface is plotted in 2-D graph. The closer the values to zero, the better the solution. Besides that, from this graphical observation, the user also can define how good the distribution. However, for some case,

the solution provided by algorithms are seem equal. To differentiate them, numerical method or mathematical approach is required to do statistical analysis. That is the use of the performance metrics discussed in the previous chapter.

For GD, DMD, MOS and HV, a reference set that can accurately represent the true-Pareto-front for each problem is required. Denote that, this true-Pareto-front could be obtained for benchmark functions only, but nearly impossible for real-world application problems. In the statistical analysis, the mean value of the best solution based on 25 independent runs are utilised for the performance and non-parametric test. To verify the performance of the proposed algorithm, the mean of GD, DMD, MOS and HV of those 25 independent runs are used to compare the diversity and accuracy of the Pareto-front produced respectively. Notify that a small value of GD, DMD and MOS correspond the high accuracy and better diversity of the produced-Pareto-front compared to true-Pareto-front, while the large value of HV indicates that the solution has better accuracy and diversity in combined sense. Last but not least, TOC also put into the analysis.

In order to know the variation or consistency of the generated solution, the standard deviation (SD) are calculated. To examine SD of each performance metric, low SD indicates that the solution has less variation and the generated solution are really close to the mean value. SD also indicates the robustness of the algorithm to find the solution. To see whether the algorithm able to reach optimum point within the search area, the best and worst of all performance metrics are also recorded.

4.2 Simulation Result

4.2.1 Pareto-front

First of all, this discussion is based on the figures illustrated in Figure 4.1 to 4.10. Denote that, true-Pareto-front is however also plotted in red-line, Pareto-front by MOSDA-NS is blue-star-dot, MOSDA-A is red-hollow-round, NSGAII is green-cross and MOPSO in brown-dot. Generally, for two-dimension MO problems, each algorithm seems able to find the Pareto-front successfully. By graphical analysis, all algorithm found correct Pareto-front for Schaffer, Schaffer N2, Fonseca and Poloni with a great performance. However, for more-than-two-dimension algorithms, the results are varied.

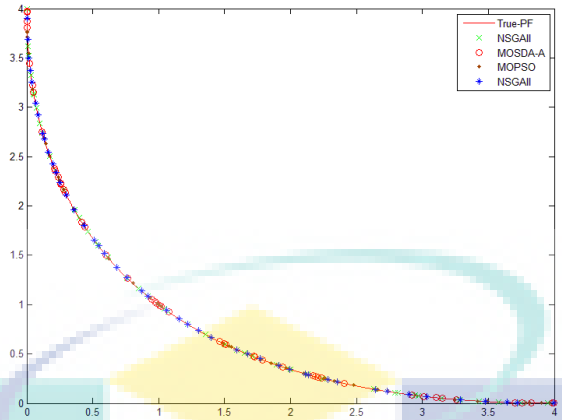

Figure 4.1 Produced-Pareto-front for f_1.

From Figure 4.1, all algorithms succeed to produce correct Pareto-front for function f_1. They managed to find all 50 POS after ran according to the desired nfe of 30,000 times. From this figure, all of the Pareto-fronts produced are almost accurate to the true- Pareto-front. Other than that, they also provide the POS in theoretical solution ranges which is $0 \leq x < 4$ and $0 \leq y < 4$. As seen, NSGAII found outermost POS on RHS while MOSDA-A found outermost POS in LHS.

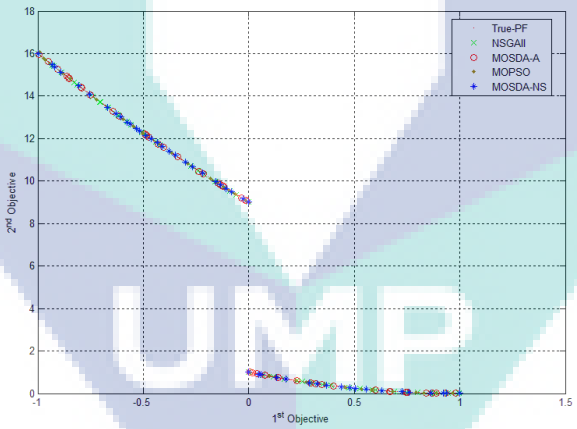

Figure 4.2 Produced-Pareto-front for f_2.

The produced Pareto-fronts for MO problem f_2 is shown in Figure 4.2. For this f_2, all algorithms succeed to find all 50 POS after ran according to desired nfe of 30,000 times. They also produced POS in which spans on the true-Pareto-front successfully. From the figure, algorithms seem obtained a fair performance of Pareto-fronts. NSGAII again provided outermost POS in RHS. At the meanwhile, MOSDA-NS provided

outermost POS in LHS. Last but not least, they also provide the POS in theoretical solution ranges which is $0 \leq x < 1$ and $0 \leq y < 16$.

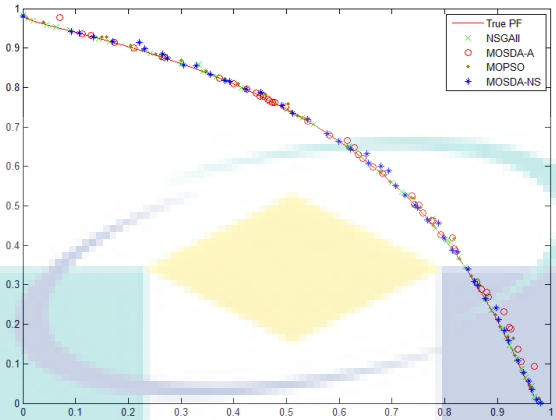

Figure 4.3 Produced-Pareto-front for f_3.

Figure 4.3 shows the Pareto-fronts produced by algorithms for f_3. By observing these produced-Pareto-fronts, all algorithms again can be concluded succeeded to produce the correct Pareto-fronts for f_3. Compared to other algorithms, NSGAII was most superior as most of the POS it produced are located on the true-Pareto-front. In contra, some of POS produced by MOSDA-NS is located a little bit far from the true-Pareto-front. In terms of diversity, both MOSDA-NS and NSGAII provide the outermost POS from both RHS and LHS.

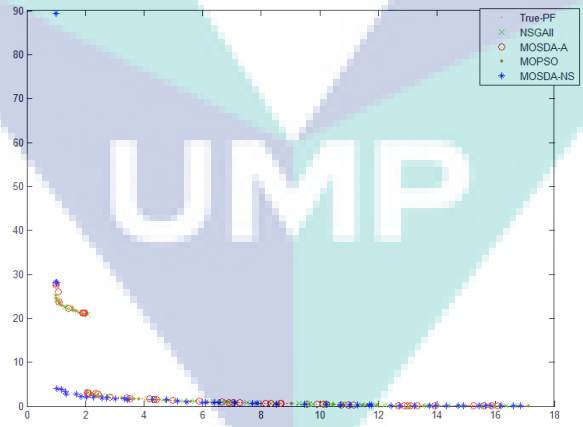

Figure 4.4 Produced-Pareto-front for f_4.

The next Figure 4.4 shows the comparison of Pareto-front-produced for function f_4. From this figure, all algorithms managed to produce Pareto-fronts same with the true-Pareto-front. In term of accuracy, all algorithms excluding MOSDA-NS manage to produce POS on the specified true-Pareto-front. However, compared to others, some POS provided by MOSDA-NS between solution ranges of $0 \leq x < 2$ and $0 \leq y < 30$ is

not on the true-Pareto-front. By observing this figure, it also can be seen that MOPSO provided outermost POS on RHS and MOSDA-NS provided outermost POS on LHS.

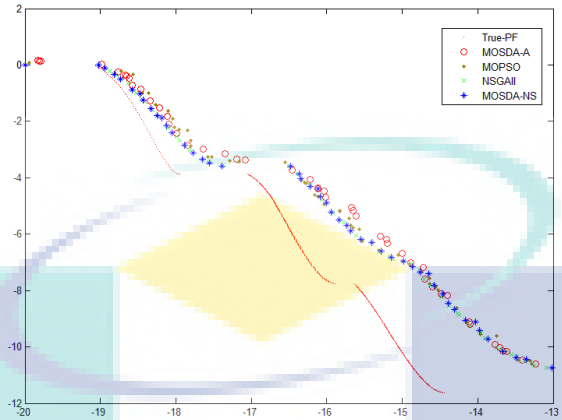

Figure 4.5 Produced-Pareto-front for f_5

Function f_5 is a three-dimensional MO problem and all Pareto-front solutions provided by all algorithms were illustrated in Figure 4.5. From the figure, all algorithms did not succeed to find accurate Pareto-fronts for f_5 even though the nfe for this function is increased to 100,000 times. They converged a little far away from the true-Pareto-front. Their Pareto-fronts also found out-of-range compared to the theoretical solution. However, to make comparison, MOSDA-NS provided most reliable Pareto-front as it is the most near to the true-Pareto-front. In contradiction, by observing the figure, MOSDA-A provided worst Pareto-front in term of accuracy and diversity.

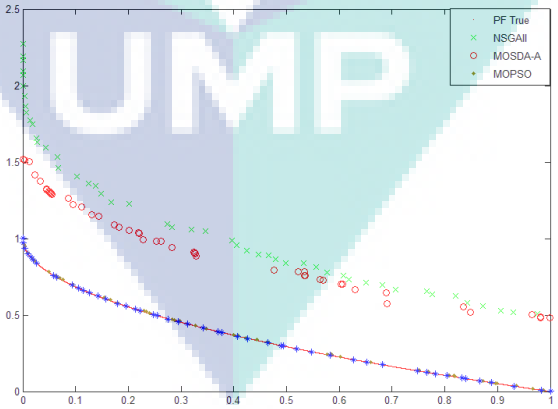

Figure 4.6 Produced-Pareto-front for f_6.

Figure 4.6 shows the solutions for function f_6. From the figure, it is clear to see that MOSDA-NS and MOPSO provided the best Pareto-front compared to others. This is because all of the POS found by them are located on the true-Pareto-front. MOSDA-NS however is a quite better than MOPSO by providing the outermost POS for both RHS and LHS. In contrast to both MOSDA-NS and MOPSO, NSGAII and MOSDA-A however could not find better solution at this particular nfe of 30,000 times.

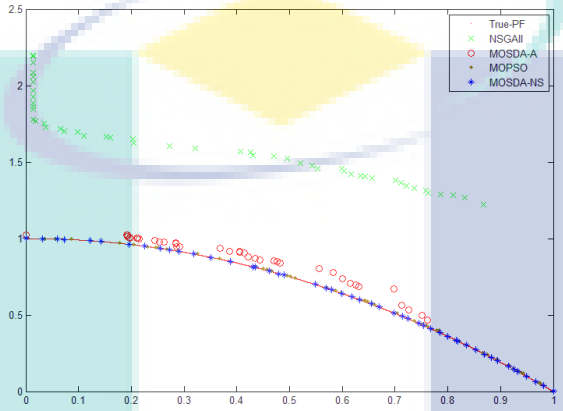

Figure 4.7 Produced-Pareto-front for f_7.

Figure 4.7 shows the Pareto-fronts produced for function f_7. By observing this figure, not all algorithm could provide the Pareto-fronts for the nfe of 30,000 times. both MOSDA-NS and MOPSO were again produced the best Pareto-fronts compared to others. Their POS were placed on the true-Pareto-front and at the correct ranges of theoretical solution. To emphasize, NSGAII again failed to provide better solution at the particular nfe of 30,000 times for this f_7. MOSDA-A in the meanwhile, produced better accurate solution compared to NSGAII however, did not better in term of diversity.

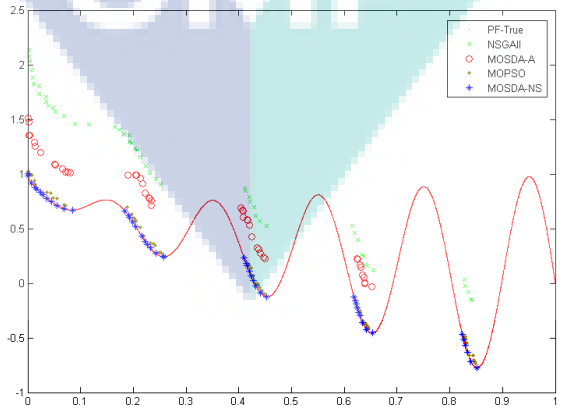

Figure 4.8 Produced-Pareto-front for f_8.

Function f_8 in which evaluated in 30,000 of nfe. Figure 4.8 shows the produced Pareto-fronts. At the first glance, MOSDA-NS produced the best Pareto-front compared to others. MOPSO went second, MOSDA-A is third and worst is NSGAII. Denote that, NSGAII required more nfe to find better solution. By looking at the figure, all algorithms succeed to find Pareto-front correctly in between $0 \leq x < 1$. However, they differed in y ranges, which theoretically is between $0 < y \leq 1$.

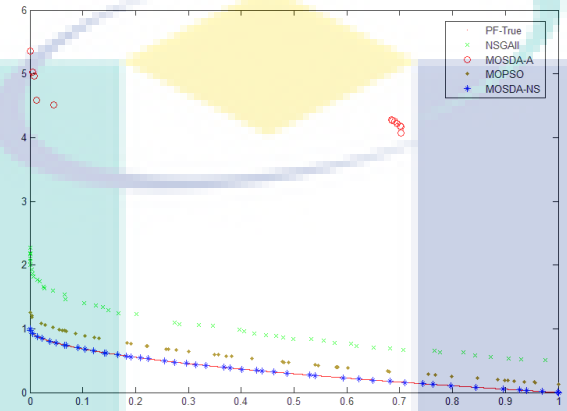

Figure 4.9 Produced-Pareto-front for f_9.

For function f_9, the produced-Pareto-fronts were plotted in Figure 4.9. From this figure, it is clear that MOSDA-NS is most superior by providing the best Pareto-front in of accuracy and diversity. All POS produced by MOSDA-NS are located on the true-Pareto-front. Clearly MOPSO and NSGAII went to second and third best Pareto-fronts while MOSDA-A was looking worst as it failed to find the correct Pareto-front. From the figure, MOSDA-A even failed to find all 50 POS and did not moved after certain particular iteration of the algorithm.

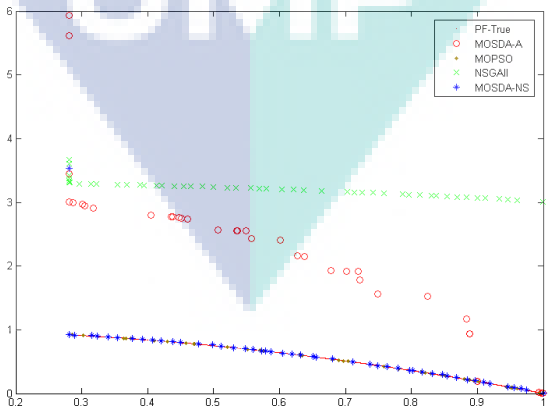

Figure 4.10 Produced-Pareto-front for f_{10}.

For the last function f_{10}, the produced-Pareto-fronts are plotted in Figure 4.10. To discuss, MOSDA-NS and MOPSO were the best Pareto-front provider for this MOP. Most of their POS are located in theoretical ranges and true-Pareto-front. However, MOSDA-A and NSGAII did not able to find the correct Pareto-fronts for this f_{10}. MOSDA-A however better in accuracy but not better in of diversity, at the meanwhile, Pareto-front by NSGAII contradicted compared to MOSDA-A.

4.2.1.1 Pareto-front Analysis

To discuss using the figures, MOSDA-NS are better than all of the compared algorithms. It produced Pareto-fronts with better accuracy and diversity. It can be seen when it produced better 7 Pareto-fronts out of 10 Pareto-fronts. MOSDA-NS outperformed other algorithms for f_1, f_2, f_6 to f_{10}. To compare MOSDA-NS against NSGAII, MOSDA-NS is better. For explanation, NS approach works well with SDA. Compare to GA, SDA is better in convergence speed. Therefore, MOSDA-NS could provide faster convergence compared to NSGAII. That is why MOSDA-NS provide better solution in the specified nfe.

To discuss the performance of MOSDA-A by observing produced-Pareto-fronts for ZDTs, it is not better than its predecessor MOPSO. In ZDT1, ZDT2, ZDT3 and ZDT6, MOSDA-A however seems better in accuracy compared to NSGAII. This due to strategy of SDA which is provide better in convergence speed compared to GA. MOSDA-A at the meanwhile do not better in spread of Pareto-front compared to NSGAII. The strategy of archive-technique not preserve the POS better than NS. The different is in strategy of determine the front for each POS by calculating $CD.CD$ proved that it can preserve the location of POS better than grid generation strategy in Archiving-method (AM). Furthermore, the POS stored in archive-technique not based on the best spread but only based on the best cost value. In contrast, NS proved that its strategy to store POS with best cost value alongside with the better spread of location.

By observing Pareto-front of ZDT4, obviously MOSDA-A failed to find the correct Pareto-front. The problem of getting trapped in local optima is occurred in solving this MOP. Throughout the test, the agents are observed stuck at the remote area. This happen at the middle of searching process. This problem could not be solved by adding

the NFE. However, the strategy to create more random of agents is really useful to avoid local optima.

4.2.2 Numerical Result and Analysis

On the other hand, in order to make numerical analysis, all performances metrics calculation was recorded. The results are stated in Table 4.1 to 4.4 of *Appendix A*. In these results, the mean value of the best solution is utilised to conduct the performance test. This mean is based on 25 independent runs.

Important to denote that, *"Best Solution"* is the highest-rank solution provided after all runs, *"Mean Solution"* is the average value, *"Worst Solution"* is the lowest-rank solution. On the other side, standard deviation was also calculated in order to know consistency of the algorithms to provide result throughout the simulation. The smaller value of all of these indicates that the solution is better.

Table 4.1 Mean and SD for GD Test (Accuracy)

	GD							
	MOSDA-A		MOSDA-NS		MOPSO		NSGAII	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
f_1	0.00003	0.00000	**0.00003**	0.00003	0.38626	1.08108	**0.00003**	**0.00000**
f_2	0.01305	0.01126	**0.00635**	0.00469	7.13371	4.06152	0.00671	**0.00106**
f_3	0.00671	0.00126	0.00384	0.00217	0.16047	0.02040	**0.00298**	**0.00066**
f_4	0.22939	0.08960	0.07212	0.01901	10.01847	1.79283	**0.04132**	**0.01744**
f_5	0.89162	0.05604	0.87125	0.84445	3.69285	0.79187	**0.83889**	**0.02631**
f_6	0.23244	0.06262	**0.00020**	**0.00005**	0.01101	0.00929	0.55284	0.12301
f_7	0.02981	0.00976	**0.00031**	**0.00004**	0.01317	0.01986	0.74270	0.13946
f_8	0.10767	0.05564	**0.00085**	**0.00031**	0.02258	0.01146	0.49459	0.10737
f_9	2.49985	2.19254	**0.00022**	**0.00005**	6.24840	5.82666	0.88400	0.43433
f_{10}	1.50050	0.72499	**0.51001**	**0.00037**	1.15807	1.55431	2.51095	0.17701

Table 4.1 shows all mean and standard deviation (SD) for accuracy test result of the Pareto-front produced by the algorithms. At the first glance, MOSDA-NS provide the best accuracy of Pareto-front for 7 out of 10 benchmark functions. For the consistency, MOSDA-NS is at the same level with NSGAII when the test record that they have equal numbers of the best SD. NSGAII on the other hand is the second best in providing accurate Pareto-front as it provides the best Pareto-front for the rest number of the benchmark functions. This test also shows that there is no solution is better than both MOSDA-NS and NSGAII provided by both MOSDA-A and MOPSO.

Table 4.2 Mean and SD for DMD Test (Spread/ Diversity)

DMD								
	MOSDA-A		MOSDA-NS		MOPSO		NSGAII	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
f_1	0.97613	0.00736	**0.94268**	**0.00400**	1.19029	0.08553	0.94366	0.00491
f_2	1.21670	0.04596	**0.99402**	0.04188	1.19405	0.09413	1.02598	**0.03405**
f_3	0.60920	0.05524	**0.30406**	**0.02801**	0.85320	0.06680	0.37230	0.04401
f_4	1.08396	0.04428	0.94752	0.02498	**0.80180**	0.11615	0.95251	**0.02135**
f_5	0.78320	0.05572	**0.42605**	**0.02477**	0.91350	0.14410	0.49966	0.03316
f_6	0.77948	0.05470	**0.37735**	**0.03265**	0.87686	0.15002	0.66775	0.03848
f_7	0.74892	0.06468	**0.37597**	**0.04496**	0.87204	0.10648	0.78494	0.05652
f_8	0.83737	0.04387	**0.61280**	**0.02105**	0.98714	0.10371	0.66509	0.03285
f_9	0.98647	0.06254	**0.38232**	**0.02715**	1.38231	0.45381	0.99332	0.19190
f_{10}	1.09237	0.09889	1.03046	0.32551	**0.82066**	0.16549	0.94279	**0.06461**

Table 4.2 shows all mean and SD for diversity test result of the Pareto-front produced by all algorithms. From the table, it is clear that MOSDA-NS produced best Pareto-front in term of solution spread. MOSDA-NS dominated the best solution when it outperformed other three algorithms when it provided eight best spread solution from 10 benchmark functions which are tested. Simultaneously, MOSDA-NS also provide most consistent solution among all algorithms. At the meanwhile, MOPSO only able to produce best spread for f_4 and f_{10}. NSGAII however only recorded best consistencies for some benchmark function while MOSDA-A not better than other in this DMD test.

Table 4.3 Mean and SD for MOS Test (Uniform Diversity)

MOS								
	MOSDA-A		MOSDA-NS		MOPSO		NSGAII	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
f_1	0.58060	0.08926	0.56530	0.09089	5.13189	3.81951	**0.55698**	**0.06876**
f_2	**4.39079**	0.78486	6.02684	0.78035	10.72165	5.47564	5.86543	**0.71732**
f_3	**0.06902**	0.01362	0.09709	**0.01203**	0.23703	0.02953	0.08640	0.01373
f_4	**5.76308**	1.52475	7.24592	1.13053	11.83993	2.38658	6.62243	**0.85178**
f_5	**0.93879**	0.18902	1.61812	0.18598	4.13892	1.06886	1.63236	**0.15456**
f_6	**0.07238**	0.01778	0.07310	**0.01013**	0.08927	0.01791	0.23311	0.09851
f_7	**0.05449**	0.01908	0.07776	**0.00995**	0.09144	0.01539	0.15599	0.04970
f_8	**0.18406**	**0.03923**	0.28254	0.03384	0.34392	0.07419	0.42223	0.09257
f_9	0.26551	0.45243	**0.07699**	**0.01221**	6.17681	5.22888	0.36961	0.23263
f_{10}	0.67768	0.28524	0.57464	0.26006	0.50059	0.57806	**0.22213**	**0.10535**

Table 4.3 shows the best mean and SD for MOS test. Denote that MOS is to recognize algorithms in term of uniformity of diversity. From the table, MOSDA-A produced the best uniform diversity. It outperformed other algorithm when tested for f_2 to f_8. However, the NSGAII is the most consistent to provide the uniform diversity followed by MOSDA-NS. Nevertheless, MOPSO is not better to compare with others in this MOS test.

Table 4.4 Mean and SD for HV Test (Accuracy and Diversity)

HV								
	MOSDA-A		MOSDA-NS		MOPSO		NSGAII	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
f_1	0.4272	0.0489	0.9992	**0.0028**	**0.2092**	0.1045	0.4080	0.0499
f_2	0.8248	0.0323	1.0000	**0.0000**	**0.5608**	0.1239	0.8220	0.0459
f_3	0.6876	0.0465	0.9992	**0.0028**	**0.6276**	0.0626	0.7020	0.0334
f_4	0.0008	0.0028	0.9688	0.0169	**0.0000**	**0.0000**	0.0020	0.0065
f_5	**1.0000**	**0.0000**	**1.0000**	**0.0000**	**1.0000**	**0.0000**	**1.0000**	**0.0000**
f_6	0.7380	0.0564	0.9996	**0.0020**	0.8072	0.0386	**0.7180**	0.0583
f_7	**0.6480**	0.0539	1.0000	**0.0000**	0.7140	0.0629	0.6744	0.0676
f_8	0.8012	0.0355	1.0000	**0.0000**	**0.7716**	0.0567	0.7816	0.0389
f_9	0.7880	0.1136	1.0000	**0.0000**	0.7260	0.0850	**0.6820**	0.0655
f_{10}	0.6028	0.0477	0.9944	**0.0065**	0.6584	0.1008	**0.5940**	0.0764

Table 4.4 shows the best mean and SD of HV test on the produced Pareto-front by all algorithms. Denote that, HV indicator is the test to measure both accuracy and spread in a combine sense. As a matter of fact, HV test shows that MOPSO is most superior after evaluate by the HV indicator. Interestingly, six Pareto-front produced are better compared to others. Instead of that, MOSDA-NS provide the most consistent results. NSGAII followed MOPSO providing the best HV for f_6, f_9 and f_{10}. In particular, from the table also shown that HV indicator could not evaluate the correct solution for f_5 caused by far away produced-Pareto-front from the true-Pareto-front.

Table 4.5 Mean and SD for TOC Test (Time of Computation)

TOC								
	MOSDA-A		MOSDA-NS		MOPSO		NSGAII	
	Mean	SD	Mean	SD	Mean	SD	Mean	SD
f_1	**25.71**	1.18	123.34	1.33	30.26	1.15	58.45	**0.15**
f_2	**16.82**	0.81	125.11	48.01	22.41	1.31	50.84	**0.24**
f_3	16.11	0.38	117.88	0.61	**6.17**	0.41	51.78	**0.18**
f_4	174.95	4.73	235.32	606.15	**6.13**	0.61	51.24	**0.15**
f_5	181.87	35.59	383.47	24.81	**16.07**	2.02	51.23	**0.18**
f_6	55.29	3.45	111.20	0.90	**21.39**	1.14	50.71	**0.13**
f_7	49.62	3.06	113.54	0.75	**17.41**	2.56	51.14	**0.19**
f_8	114.30	5.52	110.23	0.64	**20.13**	1.19	50.97	**0.13**
f_9	34.75	4.24	117.14	2.22	**15.59**	2.08	51.74	**0.14**
f_{10}	91.35	9.27	119.31	2.19	**17.91**	2.33	108.41	**0.34**

Last but not least, Table 4.5 shows the mean and SD of computation time TOC for the runs. Obviously, MOPSO produced the Pareto-fronts in shortest time for most benchmark functions which is f_3 to f_{10}. MOSDA-A come at second by providing the quickest Pareto-front for f_1 and f_2. On the other hand, SD calculation indicates that NSGAII recorded the most consistent period in producing Pareto-fronts. Inevitably, it is also notably that MOSDA-NS required most multiple of period compared to MOPSO.

Further statistical analysis involving Friedman and Wilcoxon tests are required to determine the significant value of the different or changes to the MOSDA-A and MOSDA-NS compared to their predecessor MOPSO and NSGAII. For instance, those tests can be conducted using parametric or non-parametric test. Parametric test is important to compare the optimization algorithm. However, the limitation of parametric test is the needs of three assumption, which are independent, normality and homoscedascity. If the case is violated, then the produced result is not accurate. The alternative is by conducting the non-parametric test, if the stated assumption is not satisfied.

4.2.3 Friedman and Wilcoxon Test

By definition, Friedman test (Levine *et.al.*, 2011) is applied to compare the performance of multiple numbers of algorithm, while Wilcoxon test (Shier, 2004) identified as the test to compare the performance of two algorithms. Friedman useful to define the ranking number of each algorithm while Wilcoxon used to measure the significant level of differences between two compared parameters. In this work, both tests are employed.

Friedman test conducted in this study based on the 0.05/5% degree of significant, 8 degree of freedom for 25 runs. The obtained result are mean rank, rank number (in bracket), two-tailed value, ρ and gamma-square, γ^2 (in bracket). For 0.95/95% confidence interval, the improvement or the difference is determined as significant, if the probability value less than 0.05. The result of this Friedman test are stated as in *Appendix B.*

First of all, all the algorithm recorderd two-tailed value, p less than 0.0001 which mean that there are significant different between all solution generated. Denotes that, MOSDA-NS provide most accurate solution among them when it performed optimization on Schaffer N2 and all ZDTs. Hence shows that its domination to provide accurate solution. Furthermore, MOSDA-NS also provide the best spread Pareto-front for all MO problems except for Poloni and ZDT6. In term of MOS, Friedman test show that MOSDA-NS is not better than other. The domination between these algorithms are quiet equal.

Wilcoxon test was conducted to test the significant improvement level of each couple algorithm for comparison (Daniel, 1990; Theodorsson-Norheim, E., (1987); Haynes, W., 2013). Table in *Appendix C* show the comparison of the between the possible couples of algorithms based on the statistical result. In the table, $R^+$is sum of the positive rank, R^- is the negative rank and p is the two-tailed value. To know the significant level of different, is by looking at this p value less than 0.05. Rather than indicate the different, this value also indicates the improvement. In the table, the result that bold are for p value greater than 0.05. Notice that, the p value of DMD test between MOPSO and MOSDA-A is greater than 0.05, therefore the performance is determined as no improvement. At the meanwhile by observing the p value of GD between MOSDA-A and MOSDA-NS is greater than 0.05, therefore there is not different between the accuracy of these both two Pareto-front or their performance are at the same level. Notice that there are a lot of significant different between NSGAII and MOSDA-NS. MOSDA-NS outperformed NSGAII in term of accuracy for f_1, f_5 and all f_6 to f_{10} while in terms of diversity, it defeated NSGAII except for f_1 and f_4. For the MOS test, MOSDA-NS also record a significant improvement level in performing all functions except f_2 and f_5. In comparison, MOSDA-NS also outperformed MOPSO in all test except for f_{10}, in which most of the test performed show that they are at the same level of performance. On the other hand, there are a lot of variation when comparing MOPSO and MOSDA-A. Overall, MOSDA-A did not show a significant improvement but some of the test show that MOSDA-A is not better than MOPSO. However, MOSDA-A could perform better for two-dimensional MOP to provide accurate Pareto-front. In comparison between NSGAII and MOSDA-A, overall can be said that MOSDA-A is comparable. Finally, for MOSDA-NS, most of the test show that it outperformed both NSGAII and MOPSO while MOSDA-A is not better than other.

4.2.4 Discussion and Analysis

First of all, this discussion will be divided into two sub-sections. First is for two-dimensional MO problem and second for more-than-two dimensional MO problems. Then, the discussion in this section will completely using tables in *Appendix D* as the reference. Denote that, brackets containing number after algorithm wrote are the ranks of the algorithms in a certain descripted performance test.

4.2.4.1 Two-dimensional Benchmark Analysis

This section outlines the discussion of all algorithms applied to solve benchmark problem or MO problem that has only two decision variables. This MO problem include f_1, f_2, f_3, and f_4. As these problems are only two dimensions, then the nfe_{max} was set only for 30,000 times.

For the first function f_1, overall MOSDA-NS has shown the best performance. GD test shows that MOSDA-NS (2nd) has lower accuracy of Pareto-front compared to NSGAII (1st). At the meanwhile, MOSDA-A is ranked at third place and MOPSO at last place. For the DMD spread test, MOSDA-NS and NSGAII were ranked at first and second place respectively with no significant difference. This signed that both of them have very slight difference of performance and can be said as comparable. For this f_1, MOSDA-A (3rd) also improved in term of POS spread when beat MOPSO (4th). Surprisingly, MOSDA-NS (1st) has provided the best Pareto-front in term of accuracy and diversity according to the HV test. Because of the complexity, MOSDA-NS (4th) also performed at the longest time period to finish all the nfe. Overall, MOSDA-A has been performing at the average levels. MOSDA-A however performed better compared to its predecessor MOPSO in all test GD, DMD, MOS and HV for this MO problem.

For f_2 function, in general MOSDA-NS outperformed all of the algorithms in term of accuracy and diversity. For GD test, MOSDA-NS and NSGAII were ranked at first and second place respectively. However, Wilcoxon test show that there is no significant difference between both them. Therefore, they are comparable. In term of spread, MOSDA-NS (1st) improved significantly compared to NSGAII (2nd). At the meanwhile, MOSDA-A (4th) recorded the worst spread but no significant difference to MOPSO in third place. The HV test also indicated that MOSDA-NS (1st) has the best performance. However, MOSDA-NS (3rd) could not provide better MOS value or uniform distribution compared to MOSDA-A (1st) and NSGAII (2nd). MOSDA-A again show that it has the average performance compared to other, however better than MOPSO for most of the tests.

The third function tested is f_3. The result show that the performance of the algorithm is varied. In GD test, NSGAII (1st) outperformed others with significant different to MOSDA-NS (2nd) at the next ranking. However, MOSDA-A (3rd) performed

better than MOPSO (4th) to provide accurate solution. For DMD and HV test, MOSDA-NS (1st) is more superior compared to NSGAII (2nd). These both tests show that MOSDA-NS provide the significant improvement of the Pareto-front compared to its predecessor NSGAII. For the uniform distribution, MOSDA-A is the best. After this f_3 test, MOSDA-A also can be concluded that it can performed better than MOPSO. In term of time, MOPSO (1st) iterated in shortest time at the meanwhile, MOSDA-NS (4th) is the worst performer.

From the table, GD test shows that NSGAII (1st) has the best accuracy. However, there are no such a significant different compared to MOSDA-NS (2nd). For the first time, MOPSO (1st) outperformed other algorithms in term of solution spread while NSGAII (2nd), MOSDA-NS(3rd) and MOSDA-A(4th). On the contrary, HV indicator shows that MOSDA-NS outperformed others. Primarily, small slight difference of GD and DMD between the algorithms cause this result, which contra to GD and DMD test for this function. The HV indicator also indicates that the result between NSGAII (2nd) and MOSDA-A(3rd) are comparable. For this f_4 MO problem, MOPSO (1st) provided the best solution of best diversity. DMD test for MOSDA-NS (2nd) and NSGAII (3rd) also define us that they have the same performance to provide the diverse Pareto-front. At the meanwhile, MOSDA-A (1st) produced the best uniform diversity, which shown in the MOS test.

Overall, in this category of MO problem, MOSDA-NS is superior compared to others. These MO problems are the simplest functions that can be applied to test the MO algorithm. The performance of the algorithms on these functions then can be a guide to solve real application which has more complex structure of problems. MOSDA-NS could provide the best Pareto-front in longest period of time. At the meanwhile, MOSDA-A is the best in MOS tests. Therefore, MOSDA-A can be concluded as has the average performance among these other three algorithms.

4.2.4.2 More-than-two Benchmark Analysis

This section outlines the discussion of all algorithms applied to the more-than-two decision variables of MO problem. For this type of functions, f_5 , f_6, f_7, f_8, f_9 and f_{10}were used.

f_5 is a three-dimensional MO problem. For GD test, NSGAII (1st), MOSDA-NS(2nd) and MOSDA-A (3rd) have the same level of performance in term of accuracy. MOPSO (4th) show the least performance compared to these three algorithms. In term of diversity, DMD test show that MOSDA-NS (1st) is the best. It has the significant different between NSGAII (2nd), MOSDA-A (3rd) and MOPSO (4th) at the ascending order respectively. For the uniform distribution, MOSDA-A (1st) provide the best Pareto-front. At the meanwhile, MOSDA-NS at the second place, with no significant level compared to third place NSGAII. HV indicator is not valid for this function because all of the algorithm scored zeros for most of the run test. For time, the MOPSO (1st) take the shortest, while MOSDA-NS (4th) is the last.

f_6 is a 30-dimensional MO problem. For GD and DMD test, MOSDA-NS (1st) provided the smallest values indicate that it provided most accurate and well spread Pareto-front. In this GD test, MOPSO (2nd) is better than MOSDA-A (3rd). Surprisingly, NSGAII could not provide better accuracy in this certain setup nfe. However, NSGAII placed at second ranking for spread DMD test. For HV test, MOPSO at the first rank, at the meanwhile, MOSDA-NS at the second place. Last but not least, MOSDA-A has provided the most uniform Pareto-front among the algorithm. Clearly, MOPSO (1st) took the fastest time to solve f_6 followed by NSGAII (2nd), MOSDA-A (3rd) and MOSDA-NS (4th).

f_7 is the seventh tested MO problem. For this problem, MOSDA-NS (1st) clearly produced the smallest value of GD and DMD indicates that it produced best accuracy and diverse Pareto-front solution. It also has the largest HV among these algorithms but no significant difference compared to MOPSO. MOPSO (2nd) stands at second rank for GD test while MOSDA-A (2nd) for DMD test. MOSDA-A however stands at the average level of the performance when defeated several times by MOPSO and NSGAII. MOSDA-A (3rd) in term of accuracy. However, as expected MOSDA-A has provided the best uniform Pareto-front after providing smallest value of MOS. Significantly, the MOSDA-NS stand at the second place for this MOS test. For TOC, MOPSO (1st) is the quickest, while MOSDA-NS (4th) took the longest time to solve the MO problem.

For the eighth MO problem is f_8. It is a 30-dimensional decision variables function. For this MO problem, MOSDA-NS (1st) solved it by providing the Pareto-front

with significantly the best accuracy, diversity and uniform distribution. MOPSO (2nd) came at second rank for GD test while NSGAII (2nd) came at second for DMD performance test. MOSDA-A provide third ranked in term of accuracy compared to its predecessor MOPSO. The best value of HV for MOSDA-NS (1st) strengthen the fact it produced the best performance among the algorithms. At the meanwhile, for HV indicator, MOSDA-A (2nd) and NSGAII (3rd) has a comparable result. MOSDA-A (3rd) also beat MOPSO for the well diverse Pareto-front.

The ninth MO problem is f_9 with 10 decision variables. For this problem, MOSDA-NS (2nd) could not beat NSGAII (1st) for the accurate Pareto-front. On the other hand, MOSDA-NS (1st) has provided the best diversity of Pareto-front and largest HV among another algorithm. Specifically, for this problem, MOSDA-NS (1st) also produced the most uniform distribution Pareto-front compared to others. At the meanwhile, MOSDA-A (4th) cannot find the solution for this problem and failed to find the right solution even though nfe has been increased. However, MOS test recorded that MOSDA-A (2nd) able to produce uniform distribution of Pareto-front after beat NSGAII (2nd) and MOPSO (3rd). However, this advantage is not meaningful as the accuracy and the graph pattern is totally out of form compared to theoretical Pareto-front.

f_{10} is another 10-dimensional MO problem tested with these all four algorithms. MOSDA-NS (1st) produced the best accuracy Pareto-front alongside the largest HV value. On the other hand, MOSDA-NS produced third ranked Pareto-front with uniform diversity. However, it cannot produce a well-diverse solution after ranked number fourth for DMD test. MOSDA-A also performed at low performance for this problem. It placed third in GD, HV and DMD test while number fourth for MOS.

These functions are categorized as more complex MO problem. MOSDA-A and MOSDA-NS are successfully performed these problems even some of the Pareto-front are cannot be found. For ZDTs' family MO problems, MOSDA-A at initial iterations are trapped in local optima. This situation can be seen when the search agents were remained stationary at the certain areas with certain stationery amount of found POS during the searching process. However, at the end of the process it can find the solution but with less accuracy and diversity. Exceptional for f_9 and f_{10}, MOSDA-A are kept to get trapped in local optima even the mutation operator is applied at 2 stages of the procedure. From the Pareto-front produced per test run, the users can observe that the search agent

are remain stationary at most points. This situation remains occurred when the algorithm adjusted with more NFE and more randomness operator inserted. This problem could be solved by adding another element of strategy to SDA. In term of computation time, MOPSO against remains at first ranks compared to others. NSGAII take the longest time for most MOP.

Table 4.6 Rank for compared algorithms based on Friedman vs Wilcoxon test. *(Note: Marks (Ranks)).*

	MOSDA-A	**MOPSO**	**MOSDA-NS**	**NSGAII**
GD	25(3)	25(3)	13(1)	21(2)
DMD	27(3)	31(4)	13(1)	20(2)
MOS	15(1)	34(4)	22(2)	24(3)
HV	19(2)	20(3)	12(1)	19(2)
TOC	24(2)	12(1)	37(4)	27(3)

Table 4.6 above shows the ranking of the algorithm according to Friedman vs Wilcoxon test. For instance, clearly MOSDA-NS has outperformed all another three algorithms for accuracy and diversity. It also indicated as the best when tested with HV indicator. Its predecessor NSGAII is ranked second for GD, HV and DMD tests. Hence, MOSDA-NS is concluded has better strategy than NSGAII. MOSDA-A and MOPSO is draw and ranked third for accuracy. However, compared to MOPSO, MOSDA-A has better Pareto-front distribution, if DMD and MOS is took into the account. MOSDA-A also iterates faster than MOSDA-NS and NSGAII. Thus, the strategy of AM is simpler compared to NS.

4.3 Application of The Proposed Algorithms

In this section, system modelling of an inverted pendulum (IP) is described and a Proportional-Derivative (PD) controller has been used to stabilize the pendulum. The system is of a classical IP consisting two moving masses. These two bodies will be controlled while sliding on a horizontal plane. In this chapter, the results of the system with PD-controller are shown. The main objective of this chapter is to optimize PD-controller using the proposed algorithms and analyse the performance of the optimization by comparing them against the PD-controller optimized by NSGAII and MOPSO respectively.

4.3.1 Inverted Pendulum

Inverted pendulum (IP) is a classic control problem which widely used in control studies around the world (Nasir, 2007). Its high-non-linearities and lack of stability make it suitable to test new-developed prototype controllers. An IP system consist of an inverted pole with a specific mass (m_{pen}) hinged at certain angle (θ_{pen}) from vertical axis on a cart with mass, M_{pen} and free to move horizontally (Torres-Pomales *et. al.*, 1996; Brisilla *et. al.*, 2015).

In year 2012, a researcher managed to balance an IP using PD-controller (Hasan, 2012). In this work, the author tested the frequency responses by proposing different proportional and derivatives gains. They also drew a root locus diagram to verify the stability. However, there are limitation in this technique. The vibration, slip of wheel and insufficient current were the problem faced by them.

The case study will be described in this thesis will be based on the IP system which setup in MATLAB-Simulink. At first glance, the modelling is according to the real hardware. The first phase of this case study is to design the mathematical model of the system.

IP is unstable. The pendulum will fall downward caused by the gravitational force acting on the pendulum mass. This pendulum needs to be kept vertically inverted upwards position. To do this, a motor connected to the pendulum need to be controlled using a closed-loop system. The feedback or input device to locate the current position of the pendulum is required. To control the pendulum at this position, the PD-controller has

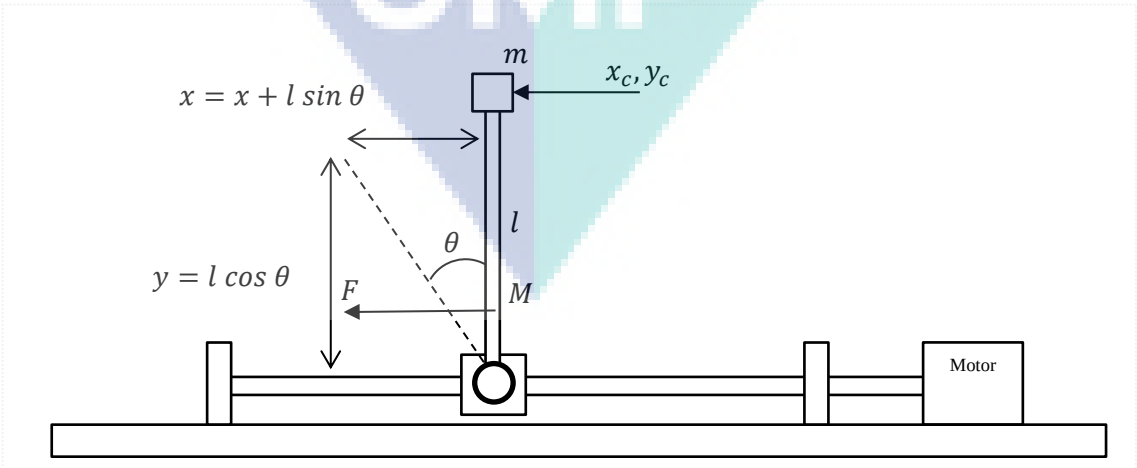

Figure 4.11 Inverted pendulum system.

been selected. This controller has been identified as the simplest control technique to apply in this research thesis.

The free-body diagram of this IP is shown in Figure 4.11. It illustrates an IP system used for the test platform in this work. It consists of a pole that can freely rotate 360 degrees and a cart that can move back and forth in a translational direction. One end of the pole is pivoted on a body of the cart while another end is left free. The pivoted pole moves linearly while the cart is in motion and at the same time, one end of the cart that is left free rotates around pivoted end of the pole. Under a static condition, the pole is pointing downward. IP is a single-input multi-output system that is normally used as a platform to test a newly developed controller. Often, a voltage is considered as the input to the system, while the linear position of the cart and the angle of the pole are considered as the outputs of the system.

Table 4.7 IP parameters and their corresponding value.

Parameter	Values
Mass of cart, M_{pen}	0.1kg
Mass of pendulum, m_{pen}	0.05kg
Friction or cart, f_{cart}	$0.1Nm^{-1}s^{-1}$
Length of pendulum, l	0.3m
Inertia of the pendulum, I	$0.006\ kg\ m^2$
Motor torque constant, K_m	$4.9\ Ncm\ A^{-1}$
Motor back emf constant, K_b	$0.0507\ V\ rad^{-1}s^{-1}$
Motor armature resistant, R	0.3Ω

4.3.2 Motion Derivation of IP

Mathematical model of the system usually presented by the Newtonian or Euler-Lagrangian equations. IP is a classical problem in dynamics and control theory. It has been used widely in experimenting the control algorithms such PD or etc.

In IP system, rotation power of the motor will be converted into straight line motion. This is done through the mechanical system of ball screw which vertically control the angle of the pendulum by the cart translation. The straight-line motion is transferred to the cart which connected to the ball screw. Therefore, the coordinate of the centroid of the pendulum are x_c and y_c. x_c and y_c are defined as in Equation 4.1 and 4.2 respectively.

$$x_c = x + l\sin\theta \quad (4.1)$$

$$y_c = l\cos\theta \quad (4.2)$$

These both centroids are used to express formula for force and moment. Therefore, Equation 4.3 and 4.4 were produce. In Equation 4.3, $\sum F$ is the summation of force while $\sum M$ is the summation of moment generated from the motor. The formulation of this force is expressed as follow:

$$\sum F : M\ddot{x} + b\dot{x} + m\ddot{x_c} = F \quad (4.3)$$

$$\sum M : m\ddot{x_c} l\cos\theta - m\ddot{y_c} l\sin\theta = mgl\sin\theta \quad (4.4)$$

From the theoretical, torque, T from motor is expressed in Equation 4.5.

$$T = K_m i \quad (4.5)$$

From Equation 4.5, i is the current that entered into the motor. The relation of T and F is expressed in Equation 4.6.

$$F = \frac{2\pi T}{r} \quad (4.6)$$

Hence, when substitute Equation 4.5 into Equation 4.6, F is expressed in Equation 4.7.

$$F = \frac{2\pi K_m i}{r} \quad (4.7)$$

At the meanwhile, the relation between voltage, V and i is as in Equation 4.8.

$$V = iR + K_b\dot{\theta} \quad (4.8)$$

The angle speed of the motor and the cart speed is expressed in Equation 4.9.

$$\dot{\theta} = \frac{2\pi}{r}\dot{x} \quad (4.9)$$

Substitute angle speed, $\dot{\theta}$ into V, then Equation 4.10 is formulated as follow.

$$V = iR + K_b\left(\frac{2\pi}{r}\dot{x}\right) \quad 4.10$$

Then, i is expressed in Equation 4.11.

$$i = \frac{V}{R} - \frac{K_b}{R}\frac{2\pi}{r}\dot{x} \quad 4.11$$

Substitute i into F, the general expression of F in Equation 4.12.

$$F = \frac{2\pi}{r}\frac{K_m}{R}\left(V - \frac{2\pi K_b}{r}\dot{x}\right) \quad 4.12$$

Supposed that $\theta = 0$. Linearized the formula F and M, then substitute the formula into Equation 4.3. Therefore Equation 4.13 is formulated.

$$(M+m)\ddot{x} + ml\ddot{\theta} + \left(b + \left(\frac{2\pi}{r}\right)^2\frac{K_m K_b}{R}\right)\dot{x} = \frac{2\pi}{r}\frac{K_m}{R}\,V \quad 4.13$$

$$m\ddot{x} + ml\,\ddot{\theta} = mg\,\theta \quad 4.14$$

F_r and F_v are formulated as in Equation 4.15.

$$F_r = b + \left(\frac{2\pi}{r}\right)^2\frac{K_m K_b}{R} \; and \; F_V = \frac{2\pi}{r}\frac{K_m}{R} \quad 4.15$$

Substitute Equation 4.15 into Equation 4.16 and 4.17, is expressed.

$$(M+m)\ddot{x} + ml\,\ddot{\theta} + F_r\dot{x} = F_V V \quad 4.16$$

$$m\ddot{x} + ml\,\ddot{\theta} = mg\,\theta \quad 4.17$$

Formulate the state space equation from the final equation. Let $x_1 = \theta, x_2 = \dot{\theta}, x_3 = \dot{x}$ and $x_4 = \ddot{x}$. Then, let $u = V$ in the formulation. From that, Equation 4.18 and 4.19 is derived.

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ \frac{(M+m)g}{Ml} & 0 & 0 & \frac{F_r}{Ml}\\ 0 & 0 & 0 & 1\\ -\frac{mg}{M} & 0 & 0 & -\frac{F_r}{m}\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0\\ -\frac{F_V}{Ml}\\ 0\\ \frac{F_V}{M}\end{bmatrix}u \quad 4.18$$

$$y = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix}$$

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ 29.4200 & 0 & 0 & 0.3090\\ 0 & 0 & 0 & 1\\ -196.1330 & 0 & 0 & -12.3615\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix} + \begin{bmatrix}0\\ -1.2121\\ 0\\ 48.4844\end{bmatrix}u \quad 4.19$$

$$y = \begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{bmatrix}$$

4.3.3 Optimization Setup

PD is the abbreviation for proportional-derivatives. PD is the type of controller which has the feedback whose output a control variable is based on the error between the user-defined set point and some gained or measured process variable (A.N.K, 2007). The block diagram shown in the Figure 4.12 below.

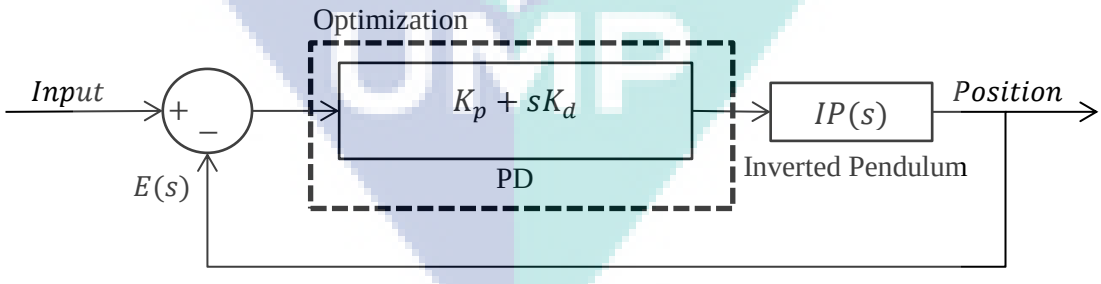

Figure 4.12 Block diagram PD system.

$$e_a(t) = K_p e(t) + K_d \frac{d_e(t)}{dt} \quad 4.20$$

From the figure, the PD control will be optimized by algorithm is shown. The formulation of PD control is stated as in Equation 4.20. At the meanwhile, the illustrated IP system will be represented by the mathematical derivation from Equation 4.18 and 4.19. These objective functions of the system are expressed in terms of the minimum mean square error (MSE) of the linear and angular velocities respectively as in Equation 4.21 and 4.22.

$$f_{11} = C_{MSE} = \min\left[\frac{1}{N}\sum_{i=1}^{N}(C_{act} - C_{des})^2\right] \quad 4.21$$

$$f_{12} = P_{MSE} = \min\left[\frac{1}{N}\sum_{i=1}^{N}(P_{act} - \mathrm{P}_{des})^2\right] \quad 4.22$$

From the equation 4.21 and 4.22, C is cart position, P is pendulum angle, C_{MSE} is mean-square-root of C, P_{MSE} is mean-square-root of P, C_{act} is the actual C, C_{des} is desired C, P_{act} is actual P, P_{des} is desired P, and N is the number of iterations. Important to emphasize, the algorithm will optimise these both Equation 4.21 and 4.22 simultaneously. All algorithms will be integrated into the system simulation files to optimise the overall MSE of the system for both linear and angular velocity of the IP.

The selected parameter for all algorithms is as in Table 3.7. Here, the comparison of parameters for all algorithms will be crucial to make a fair performance measurement. Most of the parameters selected are however not much differ to the parameters used for benchmark function tests because they are the best selected parameters for the algorithms. These parameters are measured by analysing the performance of the algorithms when solving the benchmark functions.

From the table, all algorithms will have same size of individuals in the population, which is $Y = 50$. For the related MOSDA-A and MOPSO, the size repository will be 50, $n_{rep} = 50$. The same number of grid is set by 7, $nGrid = 7$ while mutation rate, $mutrate = 0.1$ and leader selection pressure, $\beta = 2$. For the related MOSDA-NS and NSGAII, the crossover ratio, $p_c = 0.8$ and mutation ratio is $p_m = 0.3$. Other than that, both MOSDA-A and MOSDA-NS will use radius of step, $r = 0.5$ and $\theta = \frac{\pi}{4}$ for the spiral model, $S(r, \theta)$. For the rest of the parameters including which are used for

MOPSO, the same parameter as stated in papers are applied. These all parameters will be used to setup the algorithm for the IP control system optimization process.

The IP was simulated to move the pendulum in upright position at setup distance from the centre to evaluate its speed to achieve these desired positions. The simulation is again conducted by using MATLAB/Simulink environment and the model completely used the derivation from the previous section. The algorithms were in integrated and simulated with 30,000 times of NFE has successfully produced Pareto-front with desired number of decisions NDS within 23-40 iterations. The result will be shown at the next sub-section.

4.3.4 Result of PD-Controller Optimization

4.3.4.1 Pareto-front

Figure 4.13 shows the comparison of all Pareto-front produced by each algorithm for PD-controller optimization. From the figure, all MO algorithms succeed to find the correct Pareto-front for the PD-controller problems. The result ranges are between $0 \leq f_{11} < 1$and $0 \leq f_{12} < 2.5 \times 10^{-5}$. The legend at the top-right-corner shows the different

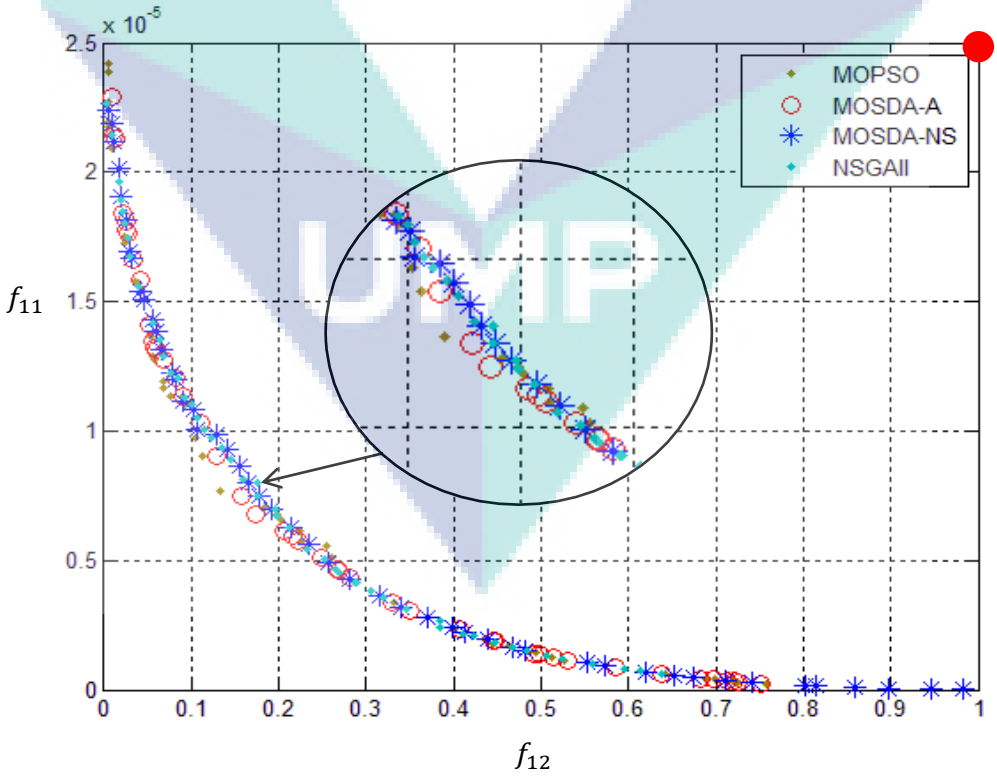

Figure 4.13 Result for PD-controller optimization by all algorithm.

marks for each algorithm. Then, the significant different of Pareto-fronts produced by algorithms are zoomed in the figure.

In this figure, MOSDA-A shows it can provide better POS as some part of them are located closer to zero. This shows that the POS is simultaneously smaller and can produces better functioning PD-controller for the IP system. Therefore, obviously can be conclude that MOSDA-A is the best algorithm to solve the optimization problem for this case study.

4.3.4.2 Numerical Result and Analysis

Simulation of IP optimization was done for 25 runs. These data were used to calculate the volume of the objective space enclosed by an approximation set and a reference point. To calculate the volume, HV test was used by using 2D reference point of $(2.5 \times 10^{-5}, 1)$. In the figure, this point is marked as red-dot. As shown in Figure 4.13, this reference point was chosen because the outermost POS found were located at these points. Table 4.8 below shows the mean, best, worst and standard deviation of all 25 simulation runs for IP problem.

Table 4.8 Mean, best, worst and SD result for HV Test (*Accuracy and Diversity*)

	MOSDA-A	**MOSDA-NS**	**NSGAII**	**MOPSO**
Mean	**0.4272**	0.4112	0.408	0.2092
Best	**0.53**	0.48	0.52	0.44
Worst	0.34	0.34	0.29	**0.03**
SD	0.048	**0.034**	0.049	0.104

From the table, the best performer was written in bold and MOSDA-A provide best mean for the problem. Other than that, MOSDA-A also provide the largest volume for this MO problem. On the other hand, the worst volume was produced by MOPSO and MOSDA-NS produced most consistent volume followed by MOSDA-A. Nevertheless, MOSDA-NS at the second place for the best mean.

4.3.4.3 Close-loop Response

At the meanwhile, this section also presents the result of the performance test performed by all algorithms to optimize PD controller for the IP. Pareto-front graph produced by these four algorithms are shown in Figures 4.14.

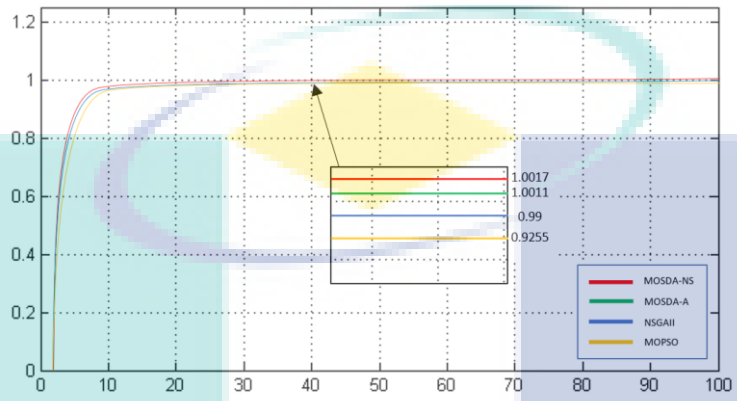

Figure 4.14 Closed-loop response produced by all optimized parameter of algorithms.

Justified that all of the algorithms managed to find the Pareto-front solution for the parameters of the PD-controller. In term of performance, all algorithms provide almost the similar accuracy and diversity. From the figure, the smoothed-blue line represents the angle and dotted-purple line represent the cart position. As the figure shown, all algorithms succeed to optimize the PD controller for the IP system.

By this figure, it can be concluded that graphically MOSDA-A provide best performance of PD optimization for the IP system, followed by MOSDA-NS. MOSDA-NS however is slower than MOSDA-A. PD which optimized by MOPSO at the meanwhile cannot achieve the desired position till 23rd seconds of the simulation. For instance, NSGAII optimized products achieved the desired position but not stable at a long time.

However, the further analysis should be constructed in order to find the differences numerically. The result of the analysis is done at stated in the Table 4.9 below.

Table 4.9 Performance of algorithms to optimize PD-controller.

	MOSDA-A	MOSDA-NS	NSGAII	MOPSO
Steady state error	**0.0011**	0.0017	0.01	0.0745
% of steady state error	**0.11**	0.17	1	7.45
Rising time	**0.27**	0.31	0.3	1.27
Settling time	35.5	**35.3**	73	83

For steady state analysis, MOSDA-A and MOSDA-NS produced best optimized-PD-controller. The percentages are the smallest among all algorithms with 0.11% and 0.17 for MOSDA-A and MOSDA-NS respectively, compared to -1% and 7.45% for both NSGAII and MOPSO respectively.

Analysis also shows that MOSDA-A could provide the optimized value to make fastest response of PD-controller. It took only 0.27 seconds to achieve desired position. There is only slightly different of rising time between MOSDA-NS and NSGAII. Therefore, both of them can be said has comparable performance in this term.

For the fourth analysis, MOSDA-NS and MOSDA-A both settled the desired position faster than NSGAII and MOPSO. This analysis how stable the optimized-PD-controller provided by both MOSDAs. NSGAII (73secs) and MOPSO (83 seconds) took very long time to stable.

4.4 Overall Performance of Proposed Algorithms

To conclude SDA, it is a deterministic type of metaheuristic algorithm. It is deterministic due to the movement of the search agents within the search space and its performance determined by the parameter's radius, r and angular displacement, θ. Denote that all of the r and θ set, all of the search agents will be end at a location. For clearer explanation, SDA adapting the swarm behaviour during the process. Rather than that, these both parameters also affect how far the search agents stepped and located from the centre, at the end of each iteration. All search agents in SDA will settle at a point at certain number of iterations. However, this not solved even by further adding the iteration number. Therefore, the algorithm will not get any better in converging to the higher accuracy of solution. This characteristic also affects the performance of its extended MOSDA-NS and MOSDA-A as it can be seen in the benchmark functions test. Then, the algorithm may be inferior in this term. The solution provided by SDA may be not global

optimum instead it is trapped in local optima. Another aspect, SDA performs spiral model involving composition of matrix, $b \times c$. Thus, as the number of variables increased, the spiral model become larger and the computation time will also increase.

For overview in term of TOC, MOSDA-NS need the longest time to iterates the solution for most MO problem. The complex structure of MOSDA-NS lead this to happen. However, for most of the MO problem, MOSDA-NS can produce better Pareto-front with the better performance. MOSDA-A seems performed at the average level. To emphasize, MOSDA-A can beat NSGAII in term of time of iteration. MOSDA-A also can be concluded as superior than its predecessor MOPSO after tested with these MO problem. Overall, for this stage of research, SDA-based version of MO algorithm is successfully developed. Through a study of existing MOPSO and NSGAII, a general descriptive framework for MO-type SDA were developed, called MOSDA-NS and MOSDA-A respectively. MOSDA-A developed incorporating the AM from MOPSO while MOSDA-NS incorporating NS method from NSGAII. AM and NS method had also been proven to be efficient in maintaining the accuracy and diversity of the Pareto-front produced, even in small population of search agents.

To test these algorithms, a benchmark study was conducted that compared the performance of four multi-objective optimization algorithms on a set of standard test problems, called the Schaffer, Poloni, Kursawe and ZDTs-based functions. All of the algorithms studied are direct methods and have some common characteristics. However, some of other aspects of these methods are significantly different. To validate the performance of these four algorithms, an empirical analysis of these novel implementations was presented using the stated benchmark functions. For revision, these benchmark functions are chosen since they demonstrate obviously the advantages and disadvantages of the randomization or mutation operator, which included in the developed algorithms structure. In each of the benchmark function test, MOSDA-A outperforms NSGAII and MOPSO in term of uniform distribution while and MOSDA-NS obviously had an improved performance in both diversity and accuracy. Its predecessor NSGAII at first outperforms MOSDA-NS in time taken to iterate and calculate. Through the empirical result also, the results indicated that the use of these two methods are improving some of the parameters while offering no substantial advantages to others.

MOSDA-NS: There is no doubt that SDA is faster than GA for SO problem. It is same going to MOSDA-NS if compared to NSGAII. At the same time of iteration term, the MOSDA-NS also parallel to MOSDA-A in which the matrix space dimension should be initialized according to the size of the MO problem. The larger the size then the longer it takes to iterate. MOSDA-NS for this stage concluded also much better than NSGAII and MOPSO. In term of accuracy, MOSDA-NS are much improved from NSGAII and MOPSO whereas it also at the higher level of hypervolume compared to NSGAII and MOPSO. It also better in diversity compared to MOPSO. For Pareto-front, MOSDA-NS able to find all of the solution and it able to provide the approximate Pareto-front and close to true-Pareto-front. Unlike MOSDA-A, MOSDA-NS able to overcome the problem of local optima for all MO problems. CD operator avoid the located POS not too gathering at a point. This operator will make sure the best-found POS alone in its location.

MOSDA-A: SDA is faster than PSO in single-objective decision making. Unlike in SO-type, MOPSO is faster than MOSDA-A in multi-objective decision making. The first thought is MOSDA-A could provide faster result than MOPSO, however the time taken is directly proportional to the size of matrix space. From observation, MOSDA-A provide a shorter time to iterate up-to-two-dimensional MO problem compared to more-than-two-dimensional MO problem. MOSDA-A however concluded as comparable to MOPSO and NSGAII in its performances. Obviously, MOSDA-A saw a marked increase in the uniform distribution but less in computational efficiency. MOSDA-A also can be seen as able to provide an approximate Pareto-front which was close to the true-Pareto-front. The algorithm could achieve this as the SDA search the solution in spiral step, which in every step it will find the better and better solutions. However, due to this strategy, MOSDA-A could have trapped in local optima, like occurred for case ZDT4. The strategy involving spiral step cause the agents intensifies in only a small region. The random strategy that adopted to solve this problem however also failed to get through this problem. The size of grid, also affect the diversity of the solution. The optimum number of this grid is really important as if it is too low then the solution would be really low in diversity and vice versa. However, if this number is too big, then it will take longer time to iterate. Hence the number of grids, which taken from the original best setup of MOPSO has provide the best dimension for grid for MOSDA-A.

Finally, the PD-controller are designed successfully in MATLAB. The swinging of the pendulum can be adjusted by manipulating the angle and position of motor shaft. The balancing of the IP is done with PD-controller. This PD-controller will perform better if the optimum constant parameters are used. The application of MOSDA-NS and MOSDA-A to optimize the controller proved that both of them are capable to defeat the superior algorithms of MOPSO and NSGAII. The simulation result was plotted and summarized in tables. Both of them provide better control of the IP compared to their predecessors. Therefore, MOSDAs is a new algorithm which still at the preliminary development, might be useful to solve problem.

CHAPTER 5

CONCLUSION

5.1 Conclusion

In Chapter 2, single-objective and multi-objective algorithm have been reviewed. The strength of SDA, MOPSO and NSGAII have been discussed as well. For now, there are some researchers who employed Archived-method (AM) and Non-dominated Sorting (NS) in MO algorithm with a great intention to maximize the potential performance of algorithms. SDA is new algorithm and yet there were only a few enhancements that have been made to SDA. Furthermore, AM and NS still have not been applied to SDA in order to make it able to deal with MO problems.

Chapter 3 explained the project flow. Other than that, MOSDA-NS and MOSDA-A are introduced by applying NS and AM in order to change the strategy of SDA to deal with MO problems. In MOSDA-NS and MOSDA-A, they create a population which consist number of individuals. These individuals then will be spread into the feasible region and the solution found will be sorted according to NS and AM respectively. Additionally, in MOSDA-NS, there will be two more populations that generated in order to give more chance in having more better POS. This chapter also discussed about all the computer simulation setup and benchmark functions.

In Chapter 4, the performance of both proposed algorithms was evaluated and discussed. These simulations involved all proposed algorithm including NSGAII and MOPSO. The data analysis from the algorithm simulations were recorded and compared to each other. Performance test measured the accuracy (GD), diversity (DMD) and uniformity of diversity (MOS) of the Pareto front that produced by all algorithms. The smaller the value that provided by the performance metric tests, the better their ability. In

this chapter also discussed about the optimization of PD-controller for IP system. The observation and analysis of this optimization have been also well described.

MOSDA-NS has shown that it is more superior than NSGAII. This is due to the best of strategy in SDA. Other than that, the better random strategy of searching the solution in feasible region has minimize the risk of search agents to fall in local optima. After comparison, MOSDA-NS has shown a great improvement in accuracy and diversity in its Pareto-front. However, the generation of solution took slightly a longer period. MOSDA-A on the other way were not much better in solving high dimension benchmark functions. However, it performed the best in optimizing PD-controller for IP system. This result shows that MOSDA-A is very dependable in solving low dimension problems, especially two dimensions.

5.2 Thesis Contribution

It is very important to denote that the first contribution of this thesis was the review about a lot of MO optimization which focused on the identification of the common to convert SO-type algorithm to MO algorithm. The aim for these reviews was to use the methodology of the existing MO algorithm, which focused on the MO algorithm that derived from SO algorithm. The methodology then summarized to use the similarities between these existing algorithms to developed SDA-based MO algorithms. This framework delivers a new perspective of existing algorithms, while allowing the improvement and developments of the new algorithms.

The second contribution was the development of two new SDA-based MO-type algorithm which named as MOSDA-NS and MOSDA-A. There is still no discovery of this gap which very important to know the performance of SDA when it applied to solve MO problem. MOSDA-A which present SDA in archiving setup provide best uniform diversity of the solution whereof the decision maker can choose the solution that fits best to their preferences. MOSDA-A however did not too consistent to solve MO problem. This consistency issue can be observed by looking at the performance of the algorithms when solving MO problem, in which some problems solved with great performance while it cannot find correct Pareto-front for other some of problems. MOSDA-NS however differ significant with MOSDA-A in which it is superior in its accuracy and diversity. Furthermore, MOSDA-NS is much improved compared to NSGAII and MOPSO in many

parameters' performance test for each MO problems. MOSDA-NS also outperformed NSGAII in terms of time taken to iterates. Thus, MOSDA-NS has improved from its NSGAII if the time to iterate is also taken into the account.

5.3 Future Works

From the conclusion, the future work that could be done to improve MOSDA is summarized as follows.

1) MOSDA-A had fell into a problem of local optima which limit its performance. However, this only happen to a MO problem, ZDT4. The presence of a hybrid or mathematical-novel method could make the SDA changes in its search strategy. To-date there very limited MO-type of SDA, then it is very widely open for researchers to do discovery in new hybrid-MOSDA.

2) Other than that, MOSDAs can be applied to test other real-world applications whichever areas or field in order to further know and measure its actual performances.

REFERENCES

Abido, M. A. (2009). Multiobjective Particle Swarm Optimization For Environmental/Economic Dispatch Problem. *Electric Power Systems Research*, *79*(7), Pp. 1105-1113.

Adelmann, A., Arbenz, P., Foster, A., & Ineichen, Y. (2015). A Homotopy Method For Large-Scale Multi-Objective Optimization. *Optimization and Control*, Cornell University Library.

Adiche, C., & Aïder, M. (2010). A Hybrid Method For Solving The Multi-Objective Assignment Problem. *Journal Of Mathematical Modelling And Algorithms*, *9*(2), 149-164.

Aïder, M. (2015). A Hybrid Method For The Multi-Objective Combinatorial Biddings Problem. U.M.B.Boumerdes, Algeria.

Algebra, L. (1980). '10.9 Simulated Annealing Methods', Cambridge University Press, Pp. 444–455.

Al-Matouq, A. (2012). Derivation Of The Maximum A Posterori Estimate For Discrete Time Descriptor Systems. *Systems and Control*, Cornell University Library.

Angus, D. (2009). Niching For Ant Colony Optimisation. In *Biologically-Inspired Optimisation Methods* (Pp. 165-188). Springer, Berlin, Heidelberg.

Ansari, E., & Zhiani Rezai, H. (2011). Solving Multi Objective Linear Bilevel Multi Follower Programming Problem. *International Journal Of Industrial Mathematics*, *3*(4), Pp. 303-316.

Antoniou, A., & Lu, W. S. (2007). The Optimization Problem (pp. 1-26). Springer US.

Arbaiy, N., & Watada, J. (2012). Multi-Objective Top-Down Decision Making Through Additive Fuzzy Goal Programming. *Sice Journal Of Control, Measurement, And System Integration*, *5*(2), Pp. 63-69.

Aronsson, H., & Huge Brodin, M. (2006). The Environmental Impact Of Changing Logistics Structures. *The International Journal Of Logistics Management*, *17*(3), Pp. 394-415.

Aziz, H., Brandl, F., & Brandt, F. (2015). Universal Pareto Dominance And Welfare For Plausible Utility Functions. *Journal Of Mathematical Economics*, *60*, Pp. 123-133.

Babu, B. V., & Anbarasu, B. (2005). Multi-Objective Differential Evolution (Mode): An Evolutionary Algorithm For Multi-Objective Optimization Problems (Moops). In *Proceedings Of International Symposium And 58th Annual Session Of Iiche*.

Bajd, T., Mihelj, M., Lenarcic, J., Stanovnik, A., & Munih, M. (2010). Robotics. Intelligent Systems, Control And Automation: Science And Engineering.

Bakhsh, A. And Hani, A. (2014) 'A Posteriori Approach For Decision-Making With Multiple Stochastic Objectives', *Life Science Journal, 2014, 11*, Pp. 1047–1052.

Bandyopadhyay, S., & Saha, S. (2012). *Unsupervised Classification: Similarity Measures, Classical And Metaheuristic Approaches, And Applications*. Springer Science & Business Media.

Bandyopadhyay, S., Saha, S., Maulik, U., & Deb, K. (2008). A Simulated Annealing-Based Multiobjective Optimization Algorithm: Amosa. *IEEE Transactions On Evolutionary Computation*, *12*(3), Pp. 269-283.

Baraskar, S. S., Banwait, S. S., & Laroiya, S. C. (2013). Multiobjective Optimization Of Electrical Discharge Machining Process Using A Hybrid Method. *Materials And Manufacturing Processes*, *28*(4), Pp. 348-354.

Barthelemy, J. F. M., & Haftka, R. T. (1993). A Fast Elitist Multiobjective Genetic Algorithm: NSGA2. *Structural Optimization*, *5*, 129–144.

Bautista, D. F. P. (2013). *Study Of Hybrid Strategies For Multi-Objective Optimization Using Gradient Based Methods And Evolutionary Algorithms* (Doctoral Dissertation, The University Of North Carolina At Charlotte).

Bechikh, S., Said, L. B., & Ghédira, K. (2011). Searching For Knee Regions Of The Pareto Front Using Mobile Reference Points. *Soft Computing*, *15*(9), Pp.1807-1823.

Benasla, L., Belmadani, A., & Rahli, M. (2014). Spiral Optimization Algorithm For Solving Combined Economic And Emission Dispatch. *International Journal Of Electrical Power & Energy Systems*, *62*, Pp. 163-174.

Bérubé, J. F., Gendreau, M., & Potvin, J. Y. (2009). An Exact ϵ-Constraint Method For Bi-Objective Combinatorial Optimization Problems: Application To The Traveling Salesman Problem With Profits. *European Journal Of Operational Research*, *194*(1), Pp. 39-50.

Bi, X., & Wang, C. (2017). An Improved NSGA-III Algorithm Based On Objective Space Decomposition For Many-Objective Optimization. *Soft Computing*, *21*(15), Pp. 4269-4296.

Bialas, W. F., & Karwan, M. H. (1980). Multilevel Optimization: A Mathematical Programming Perspective. In *Decision And Control Including The Symposium On Adaptive Processes, 1980 19th IEEE Conference On* (Vol. 19, Pp. 761-765).

Bialas, W. F., & Karwan, M. H. (1984). Two-Level Linear Programming. *Management Science*, *30*(8), Pp. 1004-1020.

Bodenhofer, U. (2003). Genetic Algorithms: Theory And Applications. *Johannes: Johannes Kepler University.* (Lecture).

Borhanifar, Z. And Shadkam, E. (2015) 'The New Hybrid Coaw Method For Solving Multi-Objective Problems', *International Journal In Foundations Of Computer Science & Technology, 5(6),* Pp. 15–22.

Bradstreet, L. (2011). *The Hypervolume Indicator For Multi-Objective Optimisation: Calculation And Use*. University Of Western Australia.

Brisilla, R. M., & Sankaranarayanan, V. (2015). Nonlinear Control Of Mobile Inverted Pendulum. *Robotics And Autonomous Systems*, *70*, Pp. 145-155.

C.L., C. And Chang, M. H. (1993) 'An Enhanced Genetic Algorithm', *In Eufit'93*, Pp. 1105–1109.

Cao, Y., Smucker, B. J., & Robinson, T. J. (2015). On Using The Hypervolume Indicator To Compare Pareto Fronts: Applications To Multi-Criteria Optimal Experimental Design. *Journal Of Statistical Planning And Inference*, *160*, 60-74.

Carr, J. (2014). An Introduction To Genetic Algorithms. *Senior Project*, pp. 1-40.

Carvalho, A. L. D. (2016). *A Hybrid Input-Output Multi-Objective Model To Assess Economic-Energy-Environment Trade-Offs: An Application To Brazil And Prospective Sugarcane Bioethanol Technologies* (Doctoral Dissertation).

Castro-Gutiérrez, J., Landa-Silva, D., & Moreno-Pérez, J. (2009). Dynamic Lexicographic Approach For Heuristic Multi-Objective Optimization. In *Proceedings Of The Workshop On Intelligent Metaheuristics For Logistic Planning (Caepia-Ttia 2009)(Seville (Spain))* (Pp. 153-163).

Ceberio, J., Mendiburu, A., & Lozano, J. A. (2015). The Linear Ordering Problem Revisited. *European Journal Of Operational Research*, *241*(3), Pp. 686-696.

Chakravarty, S., Mittra, R., & Williams, N. R. (2002). Application Of A Microgenetic Algorithm (Mga) To The Design Of Broadband Microwave Absorbers Using Multiple Frequency Selective Surface Screens Buried In Dielectrics. *IEEE Transactions On Antennas And Propagation*, *50*(3), Pp. 284-296.

Chand, S., & Wagner, M. (2015). Evolutionary many-objective optimization: A quick-start guide. *Surveys in Operations Research and Management Science*, *20*(2), 35–42.

Charwand, M., Ahmadi, A., Heidari, A. R., & Nezhad, A. E. (2015). Benders Decomposition And Normal Boundary Intersection Method For Multiobjective Decision Making Framework For An Electricity Retailer In Energy Markets. *IEEE Systems Journal*, *9*(4), Pp. 1475-1484.

Chase, N., Rademacher, M., Goodman, E., Averill, R., & Sidhu, R. (2009). A Benchmark Study Of Multi-Objective Optimization Methods. *Bmk-3021, Rev, 6*.

Chasnov, J. R. (2009). Introduction To Differential Equations. *The Hong Kong University Of Science And Technology, Department Of Mathematics, 2012*.

Cheng, H., Huang, W., Zhou, Q., & Cai, J. (2013). Solving Fuzzy Multi-Objective Linear Programming Problems Using Deviation Degree Measures And Weighted Max–Min Method. *Applied Mathematical Modelling*, *37*(10-11), Pp. 6855-6869.

Chiang, M. (2007). ELE539A: Optimization Of Communication Systems Lecture 16: Pareto Optimization And Nonconvex Optimization.

Chum, O., & Matas, J. (2005). Matching With Prosac-Progressive Sample Consensus. In *Computer Vision And Pattern Recognition, 2005. Cvpr 2005. IEEE Computer Society Conference On* (Vol. 1, Pp. 220-226).

Coello, C. C. (2006). Evolutionary Multi-Objective Optimization: A Historical View Of The Field. *IEEE Computational Intelligence Magazine*, *1*(1), 28-36.

Coello, C. C. A., & Lechuga, M. S. (2002) MOPSO: A Proposal For Multiple Objective Particle Swarm Optimization. In *Proc., Evolutionary Computation, 2002. Cec'02. Proceedings Of The 2002 Congress On* (Pp. 1051-1056).

Colledani, M., Bolognese, L., Ceglarek, D., Franchini, F., Marine, C., & Mistry, A. (2015). Multi-Objective Early-Stage Design Of Automotive Hybrid Assembly Lines. *Procedia Cirp*, *28*, 125-130.

Corne, D. W., Jerram, N. R., Knowles, J. D., & Oates, M. J. (2001). PESA-II: Region-Based Selection In Evolutionary Multiobjective Optimization. In *Proceedings Of The 3rd Annual Conference On Genetic And Evolutionary Computation* (Pp. 283-290). Morgan Kaufmann Publishers Inc.

Corne, D. W., Knowles, J. D., & Oates, M. J. (2000). The Pareto Envelope-Based Selection Algorithm For Multiobjective Optimization. In *International Conference On Parallel Problem Solving From Nature* (Pp. 839-848). Springer, Berlin, Heidelberg.

Corucci, F., Calisti, M., Hauser, H., & Laschi, C. (2015). Novelty-Based Evolutionary Design Of Morphing Underwater Robots. In *Proceedings Of The 2015 Annual Conference On Genetic And Evolutionary Computation* (Pp. 145-152). ACM.

Cyprian, I., Oyeka, A., & Ebuh, G. U. (2015). Modified Wilcoxon Signed-Rank Test, Lecture.

Daniel, W. W. (1990). Friedman Two-Way Analysis Of Variance By Ranks. *Applied Nonparametric Statistics*, Pp. 262-274.

David, K. (2005). A Brief Introduction To Neural Networks. Retrieved from http://www.dkriesel.com/en/science/neural_networks.

De Almeida, A. T., Ekenberg, L., Geiger, M. J., Leyva Lopez, J. C., & Morais, D. (2016). Building Mathematical Models For Multicriteria And Multiobjective Applications. *Mathematical Problems In Engineering*, *2016*, Pp. 2–4.

Deb, K. (2004). Single And Multi-Objective Optimization Using Evolutionary Computation. In *Hydroinformatics: (In 2 Volumes, With CD-ROM)* (Pp. 14-35).

Deb, K. (2014). Multi-Objective Optimization. In *Search Methodologies* (Pp. 403-449). Springer, Boston, MA.

Deb, K. (2014). Multi-Objective Optimization. Search Methodologies. *New York, Us: Springer*, 403-49.

Deb, K., & Bhaskara, U. (2005). Investigating Predator-Prey Algorithms For Multi-Objective Optimization. *Department Of Mechanical Engineering Indian Institute Of Technology Kanpur*.

Deb, K., & Kumar, A. (2007). Interactive Evolutionary Multi-Objective Optimization And Decision-Making Using Reference Direction Method. In *Proceedings Of The 9th Annual Conference On Genetic And Evolutionary Computation* (Pp. 781-788). ACM.

Deb, K., Agrawal, S., Pratap, A., & Meyarivan, T. (2000). A Fast Elitist Non-Dominated Sorting Genetic Algorithm For Multi-Objective Optimization: NSGA-II. In *International Conference On Parallel Problem Solving From Nature* (Pp. 849-858). Springer, Berlin, Heidelberg.

Deng, K., Zhang, J. P., & Yang, J. (2015). An Efficient Multi-Objective Community Detection Algorithm In Complex Networks. *Tehnicki Vjesnik/Technical Gazette*, *22*(2).

Dorigo, D., Prize, K., & Glover, F. (1999). An Introduction To Differential Evolution, *New Ideas In Optimization*, McGraw-Hill, pp. 79–108.

Eberhart, R., & Kennedy, J. (1995). A New Optimizer Using Particle Swarm Theory. In *Micro Machine And Human Science, 1995. Mhs'95., Proceedings Of The Sixth International Symposium On* (Pp. 39-43). IEEE.

Eiben, A. E., & Smith, J. E. (2008). Introduction To Evolutionary Computing, *Natural Computing Series*, Springer.

Eiben, A. E., & Smith, J. E. (2015). What Is An Evolutionary Algorithm?. In *Introduction To Evolutionary Computing* (Pp. 25-48). Springer, Berlin, Heidelberg.

Eichfelder, G. (2009). An Adaptive Scalarization Method In Multiobjective Optimization. *Siam Journal On Optimization*, *19*(4), Pp. 1694-1718.

El Yafrani, M., & Ahiod, B. (2016). Population-based vs. Single-solution Heuristics for the Travelling Thief Problem. *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference - GECCO '16*, (July), 317–324.

Eldos, T., & Al Qasim, R. (2013). On The Performance Of The Gravitational Search Algorithm. *International Journal Of Advanced Computer Science Applications. West Yorkshire, United Kingdom*.

Engelbrecht, A. P. (2014). Fitness function evaluations: A fair stopping condition?. In *Swarm Intelligence (SIS), 2014 IEEE Symposium on* (pp. 1-8). IEEE.

Engen, V. (2010). *Machine Learning For Network Based Intrusion Detection: An Investigation Into Discrepancies In Findings With The Kdd Cup'99 Data Set And Multi-Objective Evolution Of Neural Network Classifier Ensembles From Imbalanced Data* (Doctoral Dissertation, Bournemouth University).

Englander, J. A., Vavrina, M. A., & Ghosh, A. R. (2015). Multi-Objective Hybrid Optimal Control For Multiple-Flyby Low-Thrust Mission Design, *Astrodynamics*, NASA Goddard Space Flight Center; Greenbelt, MD United States.

Erfani, T., Utyuzhnikov, S. V., & Kolo, B. (2013). A Modified Directed Search Domain Algorithm For Multiobjective Engineering And Design Optimization. *Structural And Multidisciplinary Optimization*, *48*(6), 1129-1141.

Erickson, M., Mayer, A., & Horn, J. (2001). The Niched Pareto Genetic Algorithm 2 Applied To The Design Of Groundwater Remediation Systems. In *International Conference On Evolutionary Multi-Criterion Optimization* (Pp. 681-695). Springer, Berlin, Heidelberg.

Eskelinen, P., & Miettinen, K. (2012). Trade-Off Analysis Approach For Interactive Nonlinear Multiobjective Optimization. *Or Spectrum*, *34*(4), 803-816.

Evgeniou, T., & Pontil, M. (1999). Support Vector Machines: Theory And Applications. In *Advanced Course On Artificial Intelligence* (Pp. 249-257). Springer, Berlin, Heidelberg.

Feldman, G., Hunter, S. R., & Pasupathy, R. (2015). Multi-Objective Simulation Optimization On Finite Sets: Optimal Allocation Via Scalarization. In *Winter Simulation Conference (Wsc), 2015* (Pp. 3610-3621). IEEE.

Fischler, M. A., & Bolles, R. C. (1981). Random Sample Consensus: A Paradigm For Model Fitting With Applications To Image Analysis And Automated Cartography. *Communications Of The ACM*, *24*(6), 381-395.

Fonseca, C. M. M. D. (1995). *Multiobjective Genetic Algorithms With Application To Control Engineering Problems* (Doctoral Dissertation, University Of Sheffield).

Fonseca, C. M., & Fleming, P. J. (1995). An Overview Of Evolutionary Algorithms In Multiobjective Optimization. *Evolutionary Computation*, *3*(1), 1-16.

Gass, S. I. (1987). The Setting Of Weights In Linear Goal-Programming Problems. *Computers & Operations Research*, *14*(3), 227-229.

Ghaznavi, M. (2017). Optimality Conditions Via Scalarization For Approximate Quasi Efficiency In Multiobjective Optimization. *Filomat*, *31*(3), 671-680.

Ghiasi, H., Pasini, D., & Lessard, L. (2011). A Non-Dominated Sorting Hybrid Algorithm For Multi-Objective Optimization Of Engineering Problems. *Engineering Optimization*, *43*(1), 39-59.

Glover, F. (1990). Tabu Search: A Tutorial. *Interfaces*, *20*(4), 74-94.

Goel, T., & Deb, K. (2002). Hybrid Methods For Multi-Objective Evolutionary Algorithms. In *Proceedings Of The Fourth Asia-Pacific Conference On Simulated Evolution And Learning (Seal'02)* (Pp. 188-192).

Gorges, T. And River, Y. (2005) 'Multiple Criteria And Goal Programming', In Lindo Manual, Pp. 411–440.

Gorjestani, M., Shadkam, E., Parvizi, M., & Aminzadegan, S. (2015). A Hybrid Coa-Dea Method For Solving Multi-Objective Problems, *Optimization and Control*, Cornell University Library.

Greco, S., Figueira, J., & Ehrgott, M. (2016). *Multiple Criteria Decision Analysis*. New York: Springer.

Gupta, S., Fügenschuh, A., & Ali, I. (2018). A Multi-Criteria Goal Programming Model To Analyze The Sustainable Goals Of India. *Sustainability*, *10*(3), 778.

Gutiérrez, C., Huerga, L., & Novo, V. (2018). Nonlinear Scalarization In Multiobjective Optimization With A Polyhedral Ordering Cone. *International Transactions In Operational Research*, *25*(3), 763-779.

Hartikainen, M. (2016) 'A Posteriori Methods'. Finland: Department Of Mathematical Information Technology.

Hartikainen, M. (2016b) 'A Priori Methods In Multiobjective Optimization'. Finland: University Of Jyväskylä, Finland.

Hartikainen, M. (2016c) 'Interactive Methods In Multiobjective Optimization 2: The Details Of The Nimbus Method'. Finland: University Of Jyväskylä.

Hasan, M., Saha, C., Rahman, M. M., Sarker, M. R. I., & Aditya, S. K. (2012). Balancing Of An Inverted Pendulum Using Pd Controller. *Dhaka University Journal Of Science*, *60*(1), 115-120.

Haynes, W. (2013). Wilcoxon Rank Sum Test. In *Encyclopedia Of Systems Biology* (Pp. 2354-2355). Springer, New York, NY.

Henig, M. I., & Buchanan, J. T. (1997). Tradeoff Directions In Multiobjective Optimization Problems. *Mathematical Programming*, *78*(3), 357-374.

Hien, N. T., & Hoai, N. X. (2006). A Brief Overview Of Population Diversity Measures In Genetic Programming. In *Proc. 3rd Asian-Pacific Workshop On Genetic Programming, Hanoi, Vietnam* (Pp. 128-139).

Hine, K. (2005). *Improving Landfill Monitoring Programs With The Aid Of Geoelectrical-Imaging Techniques And Geographical Information Systems*. Chalmers Tekniska Högsk.

Hole, G. (2011) 'Wilcoxon Test Handout 2011', Sussex. Ac. UK, Pp. 1–6.

Horn, J., Nafpliotis, N., & Goldberg, D. E. (1994). A Niched Pareto Genetic Algorithm For Multiobjective Optimization. In *Proceedings Of The First IEEE Conference On Evolutionary Computation, IEEE World Congress On Computational Intelligence* (Vol. 1, Pp. 82-87).

Houck, C. R., Joines, J., & Kay, M. G. (1995). A Genetic Algorithm For Function Optimization: A Matlab Implementation. *NCSU-IE TR*, *95*(09), Pp. 1-10.

Hu, X. B., Wang, M., & Di Paolo, E. (2013). Calculating Complete And Exact Pareto Front For Multiobjective Optimization: A New Deterministic Approach For Discrete Problems. *IEEE Transactions On Cybernetics*, *43*(3), Pp. 1088-1101.

Ibrahim, A. O., Shamsuddin, S. M., Ahmad, N. B., & Salleh, M. N. M. (2014). Hybrid Nsga-Ii Of Three-Term Backpropagation Network For Multiclass Classification Problems. In *Computer And Information Sciences (Iccoins), 2014 International Conference On* (Pp. 1-6). IEEE.

Ibrahim, Z., Lim, K. S., Buyamin, S., Satiman, S. N., Suib, M. H., Muhammad, B., ... & Watada, J. (2006). Mlpso: Multi-Leader Particle Swarm Optimization For Multiobjective Optimization Problems. *ARPN Journal Of Engineering*.

Ignizio, J. P. (1976). *Goal Programming And Extensions*. Lexington Books.

Ignizio, J. P. (1985). *Introduction To Linear Goal Programming*. Sage Publications.

Ismail, M. A., Deris, S., Mohamad, M. S., Isa, M. A., Abdullah, A., Remli, M. A., & Mohi-Aldeen, S. M. (2015). A Hybrid Of Optimization Method For Multiobjective Constraint Optimization Of Biochemical System Production. *Journal Of Theoretical & Applied Information Technology*, *81*(3), Pp. 502–513.

Jaimes, A. L., Martınez, S. Z., & Coello, C. A. C. (2009). An Introduction To Multiobjective Optimization Techniques. *Optimization In Polymer Processing*, 29-57.

Järvilehto, L. (2011). Pragmatic A Priori Knowledge: A Pragmatic Approach To The Nature And Object Of What Can Be Known Independently Of Experience. *Jyväskylä Studies In Education, Psychology And Social Research*, (429).

Jensen, M. T. (2003). Guiding Single-Objective Optimization Using Multi-Objective Methods. In *Workshops On Applications Of Evolutionary Computation* (Pp. 268-279). Springer, Berlin, Heidelberg.

Jia, C., & Zhu, H. (2017). An Improved Multiobjective Particle Swarm Optimization Based On Culture Algorithms. *Algorithms*, *10*(2), 46.

Jiang, P., Wang, C., Zhou, Q., Shao, X., Shu, L., & Li, X. (2016). Optimization Of Laser Welding Process Parameters Of Stainless Steel 316l Using Fem, Kriging And Nsga-Ii. *Advances In Engineering Software*, *99*, 147-160.

Jiang, S., Ong, Y. S., Zhang, J., & Feng, L. (2014). Consistencies And Contradictions Of Performance Metrics In Multiobjective Optimization. *IEEE Trans. Cybernetics*, *44*(12), 2391-2404.

Kasimoğlu, F., Amaçlı, Ç., & Verme, K. (2016). A Survey On Interactive Approaches For Multi-Objective Decision Making Problems. *Defence Sci. J*, *15*, 231-255.

Kaveh, A., & Laknejadi, K. (2011). A Novel Hybrid Charge System Search And Particle Swarm Optimization Method For Multi-Objective Optimization. *Expert Systems With Applications*, *38*(12), Pp. 15475-15488.

Kesselring, S. (2006). Topographien mobiler Möglichkeitsräume. *Qualitative Netzwerkanalyse*, *15*(1), 333–358.

Khan, K., & Sahai, A. (2013). Spiral Dynamics Optimization-Based Algorithm For Human Health Improvement. *Computer Science & Telecommunications*, *37*(1).

Kiranyaz, S., Ince, T., & Gabbouj, M. (2014). *Multidimensional Particle Swarm Optimization For Machine Learning And Pattern Recognition*. Newyork: Springer.

Knowles, J. D., Watson, R. A., & Corne, D. W. (2001). Reducing Local Optima In Single-Objective Problems By Multi-Objectivization. In *International Conference On Evolutionary Multi-Criterion Optimization* (Pp. 269-283). Springer, Berlin, Heidelberg.

Knowles, J., & Corne, D. (1999). The Pareto Archived Evolution Strategy: A New Baseline Algorithm For Pareto Multiobjective Optimisation. In *Evolutionary Computation, 1999. CEC 99. Proceedings Of The 1999 Congress On* (Vol. 1, Pp. 98-105).

Kokash, N. (2005). An Introduction To Heuristic Algorithms. Department Of Informatics And Telecommunications, University Of Trento, Italy.

Kramer, O. (2017). Genetic Algorithm Essentials, *Studies In Computational Intelligence*.

Krause, O., Glasmachers, T., & Igel, C. (2016). Multi-Objective Optimization With Unbounded Solution Sets. In *Proceedings Of Neural Information Processing Systems (Nips) Workshop On Bayesian Optimization: Black-Box Optimization And Beyond*.

Kuhn, T. S. (2012). *The Structure Of Scientific Revolutions*. University Of Chicago Press.

Kumar, V., & Minz, S. (2014). Multi-Objective Particle Swarm Optimization: An Introduction. *SMARTCR*, *4*(5), Pp. 335-353.

Kunkle, D. (2005). A Summary And Comparison Of Moea Algorithms. In *Internal Report*. College Of Computer And Information Science, Northeastern University.

Kuo, J. L., & Chang, C. J. (2010). A Composite P&O MPPT Control With Intelligent Orthogonal Particle Swarm Optimization For Steepest Gradient River Current Power Generation System. In *Advanced Intelligent Computing Theories And Applications. With Aspects Of Artificial Intelligence* (Pp. 564-571). Springer, Berlin, Heidelberg.

Kuo, J. L., & Chang, M. T. (2015). Multiobjective Design Of Turbo Injection Mode For Axial Flux Motor In Plastic Injection Molding Machine By Particle Swarm Optimization. *Mathematical Problems In Engineering*, *2015*.

Kuroda, K., Magori, H., Ichimura, T., & Yokoyama, R. (2015). A Hybrid Multi-Objective Optimization Method Considering Optimization Problems In Power Distribution Systems. *Journal Of Modern Power Systems And Clean Energy*, *3*(1), 41-50.

Lakshika, H. E. (2014). *An Exploration Of The Complexity-Fidelity Trade-Off In Multi-Agent Systems Via An Evolutionary Framework* (Doctoral Dissertation, University Of New South Wales, Canberra, Australia).

Laumanns, M., Thiele, L., Deb, K., & Zitzler, E. (2002). Combining Convergence And Diversity In Evolutionary Multiobjective Optimization. *Evolutionary Computation*, *10*(3), Pp. 263-282.

Levine, D. M., Berenson, M. L., Hrehbiel, T. C., & Stephan, D. F. (2011). Friedman Rank Test: Nonparametric Analysis For The Randomized Block Design. *Stat. Manag. Using Ms Excel. 6/E*, 1-5.

Li, H. L. (1996). An Efficient Method For Solving Linear Goal Programming Problems. *Journal Of Optimization Theory And Applications*, *90*(2), 465-469.

Li, H., & Landa-Silva, D. (2011). An Adaptive Evolutionary Multi-Objective Approach Based On Simulated Annealing. *Evolutionary Computation*, *19*(4), 561-595.

Lim, Y. I., Floquet, P., & Joulia, X. (2001). Efficient Implementation Of The Normal Boundary Intersection (Nbi) Method On Multiobjective Optimization Problems. *Industrial & Engineering Chemistry Research*, *40*(2), 648-655.

Liu, Y., & Niu, B. (2013). A Multi-Objective Particle Swarm Optimization Based On Decomposition. In *International Conference On Intelligent Computing* (Pp. 200-205). Springer, Berlin, Heidelberg.

Lopez, J., Munera, D., Diaz, D., & Abreu, S. (2018). On Integrating Population-Based Metaheuristics with Cooperative Parallelism. *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, (August), 601–608.

Lopeza, R. H., Rittob, T. G., Sampaioc, R., & De Cursid, J. E. S. (2013). A New Multiobjective Optimization Algorithm For Nonconvex Pareto Fronts And Objective Functions.

Lotfi, S., & Karimi, F. (2017). A Hybrid Moea/D-Ts For Solving Multi-Objective Problems. *Journal Of Ai And Data Mining*, *5*(2), pp. 183-195.

Lu, J., Wang, W., Zhang, Y., & Cheng, S. (2017). Multi-Objective Optimal Design Of Stand-Alone Hybrid Energy System Using Entropy Weight Method Based On Homer. *Energies*, *10*(10), 1664.

Luenberger, D. G., & Ye, Y. (1984). *Linear And Nonlinear Programming* (Vol. 2). Reading, Ma: Addison-Wesley.

Ma, X., Liu, F., Qi, Y., Wang, X., Li, L., Jiao, L., & Gong, M. (2016). A Multiobjective Evolutionary Algorithm Based On Decision Variable Analyses For Multiobjective Optimization Problems With Large-Scale Variables. *IEEE Trans. Evolutionary Computation*, *20*(2), pp. 275-298.

Mariano-Romero, C. E., Alcocer-Yamanaka, V., & Morales, E. F. (2005). Multiobjective water pinch analysis of the Cuernavaca city water distribution network. *Evolutionary Multi-Criterion Optimization*, *3410*(January), pp. 870–884.

Marques-Silva, J., Argelich, J., Graça, A., & Lynce, I. (2011). Boolean Lexicographic Optimization: Algorithms & Applications. *Annals Of Mathematics And Artificial Intelligence*, *62*(3-4), pp. 317-343.

Mazidi, M. R., Aghazadeh, M., Teshnizi, Y. A., & Mohagheghi, E. (2013). Optimal Placement Of Switching Devices In Distribution Networks Using Multi-Objective Genetic Algorithm Nsgaii. In *Electrical Engineering (ICEE), 2013 21st Iranian Conference On* (Pp. 1-6).

Merkle, D., & Middendorf, M. (2005). Swarm Intelligence. In *Search Methodologies* (Pp. 401-435). Springer, Boston, Ma.

Michalewicz, Z., & Fogel, D. B. (2013). *How To Solve It: Modern Heuristics*. Springer Science & Business Media.

Michalewicz, Z., & Hartley, S. J. (1996). Genetic Algorithms + Data Structures = Evolution Programs. *Mathematical Intelligencer*, *18*(3), 71.

Miettinen, K., Ruiz, F., & Wierzbicki, A. P. (2008). Introduction To Multiobjective Optimization: Interactive Approaches. In *Multiobjective Optimization* (Pp. 27-57). Springer, Berlin, Heidelberg.

Mnasri, S., Abbes, F., Zidi, K., & Ghedira, K. (2013). A Multi-Objective Hybrid Bcrc-Nsgaii Algorithm To Solve The Vrptw. In *Hybrid Intelligent Systems (HIS), 2013 13th International Conference On* (Pp. 60-65).

Moore, J. (1999). Application Of Particle Swarm To Multiobjective Optimization. *Technical Report*.

Murata, T., & Ishibuchi, H. (1995). Moga: Multi-Objective Genetic Algorithms. In *Evolutionary Computation, 1995., IEEE International Conference On* (Vol. 1, P. 289).

Naderi, B., & Sadeghi, H. (2012). A Multi-Objective Simulated Annealing Algorithm To Solving Flexible No-Wait Flowshop Scheduling Problems With Transportation Times. *Journal Of Optimization In Industrial Engineering*, *5*(11), 33-41.

Nain, P. K., & Deb, K. (2005). A Multi-Objective Optimization Procedure With Successive Approximate Models. *Kangal Report*, (No. 2005002).

Nakano, Y., Tachibana, M., Yamagishi, J., & Kobayashi, T. (2006). Constrained Structural Maximum A Posteriori Linear Regression For Average-Voice-Based Speech Synthesis. In *Ninth International Conference On Spoken Language Processing*.

Nasir, A. N. K., Tokhi, M. O., Sayidmarie, O., & Ismail, R. R. (2013). A Novel Adaptive Spiral Dynamic Algorithm For Global Optimization. In *2013 13th UK Workshop On Computational Intelligence (UKCI)*, pp. 334-341.

Nasir, A. N., Razak, A. A., Ismail, R. M., & Ahmad, M. A. (2018). A Hybrid Spiral-Genetic Algorithm For Global Optimization. *Journal Of Telecommunication, Electronic And Computer Engineering (JTEC)*, *10*(1-3), pp. 93-97.

Nasir, A.N.K., Taufika, R. M., Ismail, R., & Mohd Ashraf, A. (2015). Exponential-Based Spiral Dynamic Algorithm For Modelling Of A Flexible Manipulator System. *ARPN Journal Of Engineering And Applied Sciences*, *10*(23), Pp. 17450-17456.

Nebro, A. J., Durillo, J. J., Coello, C. A. C., Luna, F., & Alba, E. (2008). A Study Of Convergence Speed In Multi-Objective Metaheuristics. In *International Conference On Parallel Problem Solving From Nature* (Pp. 763-772). Springer, Berlin, Heidelberg.

Oesterle, J., Bauernhansl, T., & Amodeo, L. (2016). Hybrid Multi-Objective Optimization Method For Solving Simultaneously The Line Balancing, Equipment And Buffer Sizing Problems For Hybrid Assembly Systems. *Procedia CIRP*, *57*, pp. 416-421.

Ogryczak, W., & Śliwiński, T. (2006). On Direct Methods For Lexicographic Min-Max Optimization. In *International Conference On Computational Science And Its Applications* (Pp. 802-811). Springer, Berlin, Heidelberg.

Ogryczak, W., & Sliwinski, T. (2007). Lexicographic Max-Min Optimization For Efficient And Fair Bandwidth Allocation. In *International Network Optimization Conference (INOC)* (Pp. 1-6).

Oltean, M., Grosan, C., Abraham, A., & Koppen, M. (2005). Multiobjective Optimization Using Adaptive Pareto Archived Evolution Strategy. In *Intelligent Systems Design And Applications, 2005. ISDA'05. Proceedings. 5th International Conference On* (Pp. 558-563).

Orumie, U. C., & Ebong, D. W. (2013). An Efficient Method Of Solving Lexicographic Linear Goal Programming Problem. *International Journal Of Scientific And Research Publications*, *3*, Pp. 1-8.

Osman, M. S., Emam, O. E., & El Sayed, M. A. (2018). Solving Multi-level Multi-objective Fractional Programming Problems with Fuzzy Demands via FGP Approach. *International Journal of Applied and Computational Mathematics*, 4(1), 41.

Osuna-Enciso, V., Cuevas, E., Oliva, D., Zúñiga, V., Pérez-Cisneros, M., & Zaldívar, D. (2016). A Multiobjective Approach To Homography Estimation. *Computational Intelligence And Neuroscience*, *2016*, 1.

Ota, R. R., Dash, J. K., Ojha, A. K., & Iter, S. (2014). A Hybrid Optimization Technique For Solving Non-Linear Multi-Objective Optimization Problem, *AMO - Advanced Modelling and Optimization*, SOA University Bhubaneswar, India.

Pagani, E., & Pellegrini, L. (2009). Scalarization And Sensitivity Analysis In Vector Optimization. *The Linear Case*.

Palakonda, V., & Mallipeddi, R. (2017). Pareto Dominance-Based Algorithms with Ranking Methods for Many-Objective Optimization. *IEEE Access*, *5*, pp. 11043–11053.

Pan, A., Tian, H., Wang, L., & Wu, Q. (2016). A Decomposition-Based Unified Evolutionary Algorithm for Many-Objective Problems Using Particle Swarm Optimization, 2016.

Pardalos, P., Migdalas, A., & Pitsoulis, L. (2008). Pareto Optimality, Game Theory And Equilibria, *17*(May 2016).

Parrill, A. L. (2000). Introduction To Evolutionary Algorithms. *Evolutionary Algorithms In Molecular Design*, pp. 1-13.

Parsopoulos, K. E., & Vrahatis, M. N. (2008). Multi-Objective Particles Swarm Optimization Approaches. In *Multi-Objective Optimization In Computational Intelligence: Theory And Practice*(Pp. 20-42). IGI Global.

Pereira, D. G., Afonso, A., & Medeiros, F. M. (2015). Overview of Friedmans Test and Post-hoc Analysis. *Communications in Statistics: Simulation and Computation*, *44*(10), 2636–2653.

Piche, R. And Penttinen, A. (2009). Posteriori, Priors, And Predictive Distributions. Jyväskylä: Tampere University Of Technology.

Rahdar, M. H., Heidari, M., Ataei, A., & Choi, J. K. (2016). Modeling And Optimization Of R-717 And R-134a Ice Thermal Energy Storage Air Conditioning Systems Using NSGA-II And MOPSO Algorithms. *Applied Thermal Engineering*, *96*, 217-227.

Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). GSA: A Gravitational Search Algorithm. *Information Sciences*, *179*(13), Pp. 2232-2248.

Reyes-Sierra, M., & Coello, C. C. (2006). Multi-Objective Particle Swarm Optimizers: A Survey Of The State-Of-The-Art. *International Journal Of Computational Intelligence Research*, *2*(3), 287-308.

Rios, L. M., & Sahinidis, N. V. (2013). Derivative-Free Optimization: A Review Of Algorithms And Comparison Of Software Implementations. *Journal Of Global Optimization*, *56*(3), 1247-1293.

Riquelme, N., Von Lücken, C., & Baran, B. (2015). Performance Metrics In Multi-Objective Optimization. In *Computing Conference (CLEI), 2015 Latin American* (Pp. 1-11).

Ruotsalainen, H. E. N. R. I. (2009). *Interactive Multiobjective Optimization In Model-Based Decision Making With Applications* (Doctoral Dissertation, Ph. D. Thesis, University Of Kuopio).

Rutenbar, R. A. (1989). Simulated Annealing Algorithms: An Overview. *IEEE Circuits And Devices Magazine*, *5*(1), 19-26.

Sabri, N. M., Puteh, M., & Mahmood, M. R. (2013). A Review Of Gravitational Search Algorithm. *Int. J. Advance. Soft Comput. Appl*, *5*(3), 1-39.

Sadollah, A., Eskandar, H., & Kim, J. H. (2015). Water Cycle Algorithm For Solving Constrained Multi-Objective Optimization Problems. *Applied Soft Computing*, *27*, Pp. 279-298.

Sadollah, A., Eskandar, H., Bahreininejad, A., & Kim, J. H. (2015). Water Cycle Algorithm For Solving Multi-Objective Optimization Problems. *Soft Computing*, *19*(9), 2587-2603.

Sadollah, A., Eskandar, H., Lee, H. M., Yoo, D. G., & Kim, J. H. (2016). Water Cycle Algorithm: A Detailed Standard Code. *Softwarex*, *5*, 37-43.

Sadrabadi, M. R., & Sadjadi, S. J. (2009). A New Interactive Method To Solve Multiobjective Linear Programming Problems. *Jsea*, *2*(4), Pp. 237-247.

Savic, D. (2002) 'Single-Objective Vs. Multi-Objective Optimisation For Integrated Decision Support, In: Integrated Assessment And Decision', In Proceedings Of The First Biennial Meeting Of The International Environmental Modelling And Software Society, Pp. 7–12.

Schaffer, J. D. (1985). Multiple Objective Optimization With Vector Evaluated Genetic Algorithms. In *Proceedings Of The First International Conference On Genetic*

Algorithms And Their Applications, 1985. Lawrence Erlbaum Associates. Inc., Publishers.

Seada, H., & Deb, K. (2015). U-NSGA-III: A Unified Evolutionary Optimization Procedure For Single, Multiple, And Many Objectives: Proof-Of-Principle Results. In *International Conference On Evolutionary Multi-Criterion Optimization* (Pp. 34-49). Springer, Cham.

Sen, S., & Pal, B. B. (2013). Interval Goal Programming Approach To Multiobjective Fuzzy Goal Programming Problem With Interval Weights. *Procedia Technology*, *10*, Pp. 587-595.

Shaheed, M. H., Azad, A. K., & Tokhi, M. O. (2006). Intelligent Modelling Of Flexible Manipulator Systems. In *Climbing And Walking Robots* (Pp. 607-614). Springer, Berlin, Heidelberg.

Shen, F., & Yang, X. (2008). Optimization Algorithm Of Urban Road Traffic Signal Plan Based On Nsgaii. In *Intelligent Computation Technology And Automation (Icicta), 2008 International Conference On* (Vol. 2, Pp. 398-401).

Shieh, M. D., Li, Y., & Yang, C. C. (2017). Product Form Design Model Based On Multiobjective Optimization And Multicriteria Decision-Making. *Mathematical Problems In Engineering*, *2017*.

Shier, R. (2004). Statistics: 2.2 The Wilcoxon Signed Rank Sum Test. *Mathematics Learning Support Centre. Retrieved From Http://Www. Statstutor. Ac. Uk/Resources/Uploaded/Wilcoxonsignedranktest. Pdf*.

Shih, H. S., Lai, Y. J., & Lee, E. S. (1996). Fuzzy Approach For Multi-Level Programming Problems. *Computers & Operations Research*, *23*(1), Pp. 73-91.

Shukla, P. K. (2007). On The Normal Boundary Intersection Method For Generation Of Efficient Front. In *International Conference On Computational Science* (Pp. 310-317). Springer, Berlin, Heidelberg.

Siddiqui, S., Azarm, S., & Gabriel, S. A. (2012). On Improving Normal Boundary Intersection Method For Generation Of Pareto Frontier. *Structural And Multidisciplinary Optimization*, *46*(6), Pp. 839-852.

Sittisak, P., & Ekasingh, B. (2015). Trade-Offs Between The Economic, Social And Environmental Objectives In Optimal Resource Management In The Fang Watershed, Chiang Mai Province, Thailand. *Agriculture And Agricultural Science Procedia*, *5*, Pp. 38-45.

Sivanandam, S. N., & Deepa, S. N. (2008). Genetic Algorithm Optimization Problems. In *Introduction To Genetic Algorithms*(Pp. 165-209). Springer, Berlin, Heidelberg.

Sörensen, K., Sevaux, M., & Glover, F. (2018). A History Of Metaheuristics. *Handbook Of Heuristics*, 1-18.

Srinivas, N., & Deb, K. (1994). Muiltiobjective Optimization Using Nondominated Sorting In Genetic Algorithms. *Evolutionary Computation*, *2*(3), Pp. 221-248.

Stanford, U. (2012) 'Gradient-Free Optimization', Multidisciplinary Design Optimization, Pp. 133–161.

Storn, R., & Price, K. (1997). Differential Evolution–A Simple And Efficient Heuristic For Global Optimization Over Continuous Spaces. *Journal Of Global Optimization*, *11*(4), 341-359.

Suppapitnarm, A., Seffen, K. A., Parks, G. T., & Clarkson, P. J. (2000). A Simulated Annealing Algorithm For Multiobjective Optimization. *Engineering Optimization*, *33*(1), 59-85.

Sutradhar, S., Choudhury, N. B., & Sinha, N. (2016). Modelling Of Hydrothermal Unit Commitment Coordination Using Efficient Metaheuristic Algorithm: A Hybridized Approach. *Journal Of Optimization*, *2016*.

Szopa, R., & Marczyk, B. (2011). Optimization Of Production Problems Using Mathematical Programming. *Polish Journal Of Management Studies*, *4*, 231-238.

Tamura, K., & Yasuda, K. (2011). Primary Study Of Spiral Dynamics Inspired Optimization. *IEEJ Transactions On Electrical And Electronic Engineering*, *6*(S1), S98-S100.

Tamura, K., & Yasuda, K. (2011). Spiral Multipoint Search For Global Optimization. In *Machine Learning And Applications And Workshops (Icmla), 2011 10th International Conference On* (Vol. 1, Pp. 470-475).

Tamura, K., & Yasuda, K. (2013). A Stability Analysis Based Parameter Setting Method For Spiral Optimization. In *Systems, Man, And Cybernetics (Smc), 2013 IEEE International Conference On* (Pp. 3909-3914).

Tamura, K., & Yasuda, K. (2016). Spiral Optimization Algorithm Using Periodic Descent Directions. *SICE Journal Of Control, Measurement, And System Integration*, *9*(3), 134-143.

Tang, Y. C., & Chang, C. T. (2012). Multicriteria Decision-Making Based On Goal Programming And Fuzzy Analytic Hierarchy Process: An Application To Capital Budgeting Problem. *Knowledge-Based Systems*, *26*, 288-293.

Theodorsson-Norheim, E. (1987). Friedman And Quade Tests: Basic Computer Program To Perform Nonparametric Two-Way Analysis Of Variance And Multiple Comparisons On Ranks Of Several Related Samples. *Computers In Biology And Medicine*, *17*(2), 85-99.

Tian, H., Li, K., & Liu, W. (2016). A Pareto-Based Adaptive Variable Neighborhood Search For Biobjective Hybrid Flow Shop Scheduling Problem With Sequence-Dependent Setup Time. *Mathematical Problems In Engineering*, *2016*.

Tian, H., Yuan, X., Ji, B., & Chen, Z. (2014). Multi-Objective Optimization Of Short-Term Hydrothermal Scheduling Using Non-Dominated Sorting Gravitational Search Algorithm With Chaotic Mutation. *Energy Conversion And Management, 81*, 504-519.

Tiwari, S., Fadel, G., & Deb, K. (2011). Amga2: Improving The Performance Of The Archive-Based Micro-Genetic Algorithm For Multi-Objective Optimization. *Engineering Optimization, 43*(4), Pp. 377-401.

Tiwari, S., Koch, P., Fadel, G., & Deb, K. (2008). AMGA: An Archive-Based Micro Genetic Algorithm For Multi-Objective Optimization. In *Proceedings Of The 10th Annual Conference On Genetic And Evolutionary Computation* (Pp. 729-736). ACM.

Torres-Pomales, W., & Gonzalez, O. R. (1996). Nonlinear Control Of Swing-Up Inverted Pendulum. In *Control Applications, 1996., Proceedings Of The 1996 IEEE International Conference On* (Pp. 259-264).

Tseng, L. Y. (2003). Metaheuristic Methods And Their Applications.

Umadevi, B., Sundar, D., & Alli, D. R. P. (2014). Enhancement Of The Portfolio Determination Using Multi-Objective Optimization.

Umair F. Siddiqi, Sadiq M. Sait. (2017) A New Heuristic for the Data Clustering Problem. *IEEE Access* 5, Pp. 6801-6812

Wang, K., & Utyuzhnikov, S. V. (2017). An Extension Of The Directed Search Domain Algorithm To Bilevel Optimization. *Engineering Optimization, 49*(8), pp. 1420-1440.

Wang, K., & Utyuzhnikov, S. V. (2018). A Modified Rotation Strategy For Directed Search Domain Algorithm In Multiobjective Engineering Optimization. *Structural And Multidisciplinary Optimization, 57*(2), pp. 877-890.

Wang, R. (2016). An Improved Nondominated Sorting Genetic Algorithm for Multiobjective Problem, *2016*.

Wilde, D. J. (1962). Optimization Methods. In *Advances In Chemical Engineering* (Vol. 3, Pp. 273-332). Academic Press.

Williamson, T. (2013). Between A Priori And A Posteriori Knowledge?. *The A Priori In Philosophy*, 291.

Wolpert, D. H., & Macready, W. G. (1997). No Free Lunch Theorems For Optimization. *IEEE Transactions On Evolutionary Computation,* 1(1), 67-82.

Womersley, R. (2008) 'Local And Global Optimization', Maths.Unsw.Edu.Au. New South Wales Local: University Of New South Wales.

Wong, K. C. (2015). Evolutionary Algorithms: Concepts, Designs, And Applications In Bioinformatics: Evolutionary Algorithms For Bioinformatics. *Neural and Evolutionary Computing*, Cornell University Library.

Wu, G. (2017). A Multi-Objective Trade-Off Model In Sustainable Construction Projects. *Sustainability*, *9*(11), 1929.

Xu, B., Qi, J., Zhou, C., Hu, X., Xu, B., & Sun, Y. (2015). Hybrid Self-Adaptive Algorithm For Community Detection In Complex Networks. *Mathematical Problems In Engineering*, *2015*.

Yang, G., Xu, T., Li, X., Xiu, H., & Xu, T. (2015). An Efficient Hybrid Algorithm For Multiobjective Optimization Problems With Upper And Lower Bounds In Engineering. *Mathematical Problems In Engineering*.

Yang, X. S. (2010). *Nature-Inspired Metaheuristic Algorithms*. Luniver Press.

Yang, X. S. (2012). Nature-inspired mateheuristic algorithms: Success and new challenges. *Optimization and Control*, Cornell University Library.

Yang, X. S. (2013). Optimization and metaheuristic algorithms in engineering. In *Metaheuristics in water, geotechnical and transport engineering* (pp. 1-23).

Yang, X. S. And He, X. (2013). 'Bat Algorithm: Literature Review And Applications', International Journal Of Bio-Inspired Computation, 5(3), P. 141.

Yijie, S., & Gongzhang, S. (2008). Improved NSGA-II multi-objective genetic algorithm based on hybridization-encouraged mechanism. *Chinese Journal of Aeronautics*, *21*(6), 540-549.

Yuan, Y., Xu, H., & Wang, B. (2014). An improved NSGA-III procedure for evolutionary many-objective optimization. In *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation* (pp. 661-668). ACM.

Zelinka, J., Romportl, J., & Müller, L. (2010). A priori and a posteriori machine learning and nonlinear artificial neural networks. In *International Conference on Text, Speech and Dialogue* (pp. 472-479). Springer, Berlin, Heidelberg.

Zhang, Y., Wang, S., & Ji, G. (2015). A Comprehensive Survey on Particle Swarm Optimization Algorithm and Its Applications, 2015.

Zhao, J., Yu, M., & Wang, T. (2009). A multi-ridge recognition method using modified MOPSO algorithm and its application on modal parameter identification. In *Image and Signal Processing, 2009. CISP'09. 2nd International Congress on* (pp. 1-5).

Zhukovin, V., Chkhikvadze, N., & Alimbarashvili, Z. (2010). Decision Making: Lexicographical Procedure. *Optimization and Control*, Cornell University Library.

Zitzler, E., & Thiele, L. (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE transactions on Evolutionary Computation*, *3*(4), 257-271.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*, *8*(2), 173-195.

Zitzler, E., Laumanns, M., & Bleuler, S. (2004). A tutorial on evolutionary multiobjective optimization. In *Metaheuristics for multiobjective optimisation* (pp. 3-37). Springer, Berlin, Heidelberg.

Zitzler, E., Laumanns, M., & Thiele, L. (2001). SPEA2: Improving The Strength Pareto Evolutionary Algorithm. *Tik-Report*, *103*.

APPENDIX A
BEST, MEAN, WORST AND STANDARD DEVIATION FOR ALL ALGORITHM

Table 1 Summary of statistical optimization result obtained by MOSDA-NS for all reported MO problems.

MOSDA-NS																				
	Best Solution					Mean Solution					Worst Solution					Standard Deviation				
	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC
f_1	0.0000	0.9332	0.3603	1.00	120.28	0.0000	0.9427	0.5653	1.00	123.34	0.0000	0.9497	0.7745	0.99	125.78	0.0000	0.0040	0.0909	0.00	1.33
f_2	0.0047	0.9394	4.2139	1.00	113.99	0.0063	0.9940	6.0268	1.00	125.11	0.0094	1.0636	7.2006	1.00	355.54	0.0013	0.0419	0.7804	0.00	48.01
f_3	0.0022	0.2586	0.0741	1.00	116.99	0.0038	0.3041	0.0971	1.00	117.88	0.0066	0.3669	0.1273	0.99	119.51	0.0009	0.0280	0.0120	0.00	0.61
f_4	0.0190	0.9134	5.0725	1.00	110.18	0.0721	0.9475	7.2459	0.97	235.32	0.3339	1.0280	9.4783	0.93	3144.82	0.0795	0.0250	1.1305	0.02	606.15
f_5	0.8445	0.3775	1.2070	1.00	377.43	0.8712	0.4261	1.6181	1.00	383.47	0.9078	0.4708	2.0178	1.00	502.53	0.0170	0.0248	0.1860	0.00	24.81
f_6	0.0000	0.3293	0.0541	1.00	109.78	0.0002	0.3773	0.0731	1.00	111.20	0.0014	0.4520	0.0876	0.99	113.90	0.0003	0.0327	0.0101	0.00	0.90
f_7	0.0000	0.3058	0.0607	1.00	112.35	0.0003	0.3760	0.0778	1.00	113.54	0.0020	0.4970	0.0940	1.00	115.10	0.0006	0.0450	0.0099	0.00	0.75
f_8	0.0003	0.5712	0.2217	1.00	109.29	0.0009	0.6128	0.2825	1.00	110.23	0.0020	0.6494	0.3345	1.00	111.98	0.0004	0.0211	0.0338	0.00	0.64
f_9	0.0000	0.3224	0.0546	1.00	114.53	0.0002	0.3823	0.0770	1.00	117.14	0.0009	0.4377	0.1015	1.00	120.15	0.0003	0.0272	0.0122	0.00	2.22
f_{10}	0.0004	0.2750	0.0772	1.00	114.46	0.5100	1.0305	0.5746	0.99	119.31	0.9580	1.2819	0.9643	0.98	121.90	0.2848	0.3255	0.2601	0.01	2.19

Table 2 Summary of statistical optimization result obtained by MOSDA-A for all reported MO problems.

MOSDA-A																				
	Best Solution					Mean Solution					Worst Solution					Standard Deviation				
	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC
f_1	0.0000	0.9604	0.3936	0.53	23.77	0.0000	0.9761	0.5806	0.43	25.71	0.0000	0.9901	0.7240	0.34	26.78	0.0000	0.0074	0.0893	0.05	1.18
f_2	0.0049	1.1285	2.9710	0.89	15.90	0.0131	1.2167	4.3908	0.82	16.82	0.0484	1.3045	5.8625	0.77	18.42	0.0113	0.0460	0.7849	0.03	0.81
f_3	0.0046	0.5258	0.0447	0.79	15.28	0.0067	0.6092	0.0690	0.69	16.11	0.0097	0.7325	0.1001	0.57	16.97	0.0013	0.0552	0.0136	0.05	0.38
f_4	0.0430	1.0170	2.8245	0.01	167.06	0.2294	1.0840	5.7631	0.00	174.95	0.3851	1.2004	8.3166	0.00	185.53	0.0896	0.0443	1.5247	0.00	4.73
f_5	0.7794	0.6844	0.7190	1.00	138.08	0.8916	0.7832	0.9388	1.00	181.87	0.9703	0.9111	1.3591	1.00	284.38	0.0560	0.0557	0.1890	0.00	35.59
f_6	0.1265	0.6529	0.0369	0.84	48.27	0.2324	0.7795	0.0724	0.74	55.29	0.3580	0.8994	0.1069	0.61	60.16	0.0626	0.0547	0.0178	0.06	3.45
f_7	0.0170	0.6395	0.0215	0.73	38.89	0.0298	0.7489	0.0545	0.65	49.62	0.0627	0.8463	0.0914	0.57	52.85	0.0098	0.0647	0.0191	0.05	3.06
f_8	0.0272	0.7571	0.0905	0.85	105.92	0.1077	0.8374	0.1841	0.80	114.30	0.2494	0.9404	0.2514	0.70	127.67	0.0556	0.0439	0.0392	0.04	5.52
f_9	0.9323	0.8943	0.0151	0.99	32.03	2.4999	0.9865	0.2655	0.79	34.75	11.3244	1.1479	2.3279	0.52	47.39	2.1925	0.0625	0.4524	0.11	4.24
f_{10}	0.4727	0.9373	0.2253	0.71	68.52	1.5005	1.0924	0.6777	0.60	91.35	3.0748	1.3160	1.1995	0.52	102.48	0.7250	0.0989	0.2852	0.05	9.27

Table 3 Summary of statistical optimization result obtained by MOPSO for all reported MO problems.

MOPSO																				
	Best Solution					Mean Solution					Worst Solution					Standard Deviation				
	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC
f_1	0.0000	1.0365	0.8684	0.44	27.63	0.3863	1.1903	5.1319	0.21	30.26	4.5932	1.3644	18.0255	0.03	31.37	1.0811	0.0855	3.8195	0.10	1.15
f_2	1.2398	1.0610	3.5618	0.79	19.73	7.1337	1.1941	10.7217	0.56	22.41	20.0895	1.4885	30.3463	0.36	24.35	4.0615	0.0941	5.4756	0.12	1.31
f_3	0.1338	0.6585	0.1770	0.78	5.08	0.1605	0.8532	0.2370	0.63	6.17	0.2120	0.9630	0.2883	0.50	7.18	0.0204	0.0668	0.0295	0.06	0.41
f_4	7.1915	0.6235	7.3924	0.00	4.97	10.0185	0.8018	11.8399	0.00	6.13	13.8856	1.1619	15.6853	0.00	7.41	1.7928	0.1161	2.3866	0.00	0.61
f_5	2.1659	0.6796	2.6053	1.00	11.76	3.6929	0.9135	4.1389	1.00	16.07	5.0683	1.2302	7.2504	1.00	19.89	0.7919	0.1441	1.0689	0.00	2.02
f_6	0.0020	0.6516	0.0523	0.89	18.09	0.0110	0.8769	0.0893	0.81	21.39	0.0424	1.3649	0.1098	0.74	23.28	0.0093	0.1500	0.0179	0.04	1.14
f_7	0.0006	0.6190	0.0614	0.86	12.03	0.0132	0.8720	0.0914	0.71	17.41	0.0739	1.1222	0.1203	0.60	21.48	0.0199	0.1065	0.0154	0.06	2.56
f_8	0.0037	0.7948	0.1957	0.87	17.01	0.0226	0.9871	0.3439	0.77	20.13	0.0498	1.1563	0.5012	0.63	22.91	0.0115	0.1037	0.0742	0.06	1.19
f_9	0.0000	0.5403	0.0402	0.89	12.13	6.2484	1.3823	6.1768	0.73	15.59	19.8761	1.9147	17.4421	0.59	20.14	5.8267	0.4538	5.2289	0.09	2.08
f_{10}	0.0002	0.6309	0.0683	0.92	12.55	1.1581	0.8207	0.5006	0.66	17.91	4.9236	1.2379	2.4785	0.46	21.48	1.5543	0.1655	0.5781	0.10	2.33

Table 4 Summary of statistical optimization result obtained by NSGAII for all reported MO problems.

NSGAII																				
	Best Solution					Mean Solution					Worst Solution					Standard Deviation				
	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC	GD	DMD	MOS	HV	TOC
f_1	0.0000	0.9329	0.4422	0.52	58.12	0.0000	0.9437	0.5570	0.41	58.45	0.0000	0.9519	0.7303	0.29	58.71	0.0000	0.0049	0.0688	0.05	0.15
f_2	0.0051	0.9788	4.6006	0.93	50.47	0.0067	1.0260	5.8654	0.82	50.84	0.0091	1.0959	7.4375	0.73	51.75	0.0011	0.0340	0.7173	0.05	0.24
f_3	0.0022	0.2833	0.0588	0.76	51.35	0.0030	0.3723	0.0864	0.70	51.78	0.0046	0.4574	0.1220	0.64	52.14	0.0007	0.0440	0.0137	0.03	0.18
f_4	0.0225	0.9167	5.1002	0.03	50.85	0.0413	0.9525	6.6224	0.00	51.24	0.1088	0.9963	8.4902	0.00	51.44	0.0174	0.0213	0.8518	0.01	0.15
f_5	0.7840	0.4476	1.3100	1.00	50.78	0.8389	0.4997	1.6324	1.00	51.23	0.8947	0.5581	1.9877	1.00	51.50	0.0263	0.0332	0.1546	0.00	0.18
f_6	0.3660	0.5738	0.0901	0.84	50.46	0.5528	0.6678	0.2331	0.72	50.71	0.8088	0.7179	0.4273	0.60	51.06	0.1230	0.0385	0.0985	0.06	0.13
f_7	0.5079	0.6778	0.0859	0.84	50.86	0.7427	0.7849	0.1560	0.67	51.14	0.9849	0.9110	0.2590	0.55	51.87	0.1395	0.0565	0.0497	0.07	0.19
f_8	0.2789	0.6127	0.2669	0.86	50.76	0.4946	0.6651	0.4222	0.78	50.97	0.7063	0.7380	0.6206	0.73	51.36	0.1074	0.0329	0.0926	0.04	0.13
f_9	0.4569	0.6841	0.0335	0.84	51.47	0.8840	0.9933	0.3696	0.68	51.74	2.3735	1.6674	1.0161	0.59	51.96	0.4343	0.1919	0.2326	0.07	0.14
f_{10}	2.2114	0.8838	0.1134	0.80	107.83	2.5109	0.9428	0.2221	0.59	108.41	2.9254	1.1386	0.4939	0.49	109.09	0.1770	0.0646	0.1053	0.08	0.34

APPENDIX B
FRIEDMAN TEST

Function	Parameter		Friedman Test			
			MOPSO	MOSDA-NS	MOSDA-A	NSGAII
f_1	GD	Mean rank	3.76	2.08	2.36	**1.8(1)**
		$\rho(\gamma^2)$	0.00001(34.104)			
	DMD	Mean rank	4	**1.32(1)**	3	1.68
		$\rho(\gamma^2)$	0.00001(68.472)			
	MOS	Mean rank	4	2	2.12	**1.88(1)**
		$\rho(\gamma^2)$	0.00001(45.432)			
	HV	Mean rank	**1.04(1)**	4	2.56	2.4
		$\rho(\gamma^2)$	0.00001(66.460)			
	TOC	Mean rank	2	4	**1(1)**	3
		$\rho(\gamma^2)$	0.00001(75.000)			
f_2	GD	Mean rank	4	**1.64(1)**	2.56	1.8
		$\rho(\gamma^2)$	0.00001(52.248)			
	DMD	Mean rank	3.28	**1.36(1)**	3.68	1.68
		$\rho(\gamma^2)$	0.00001(59.592)			
	MOS	Mean rank	3.68	2.68	**1.32(1)**	2.32
		$\rho(\gamma^2)$	0.00001(42.744)			
	HV	Mean rank	**1(1)**	4	2.52	2.48
		$\rho(\gamma^2)$	0.00001(67.512)			
	TOC	Mean rank	2	4	**1(1)**	3
		$\rho(\gamma^2)$	0.00001(75.000)			
f_3	GD	Mean rank	4	1.76	2.96	**1.28(1)**
		$\rho(\gamma^2)$	0.00001(67.464)			
	DMD	Mean rank	4	**1(1)**	3	2
		$\rho(\gamma^2)$	0.00001(75.000)			
	MOS	Mean rank	4	2.6	**1.28(1)**	2.12
		$\rho(\gamma^2)$	0.00001(58.392)			
	HV	Mean rank	**1.32(1)**	4	2.24	2.44
		$\rho(\gamma^2)$	0.00001(56.841)			
	TOC	Mean rank	**1(1)**	4	2	3
		$\rho(\gamma^2)$	0.00001(75.000)			
f_4	GD	Mean rank	4	1.68	2.88	**1.44(1)**
		$\rho(\gamma^2)$	0.00001(62.856)			
	DMD	Mean rank	**1.2(1)**	2.48	3.96	2.36
		$\rho(\gamma^2)$	0.00001(57.624)			
	MOS	Mean rank	3.88	2.64	**1.52(1)**	1.96
		$\rho(\gamma^2)$	0.00001(68.564)			
	HV	Mean rank	**1.9(1)**	4	2.02	2.08
		$\rho(\gamma^2)$	0.00001(68.564)			
	TOC	Mean rank	**1(1)**	3.04	3.96	2
		$\rho(\gamma^2)$	0.00001(73.848)			

Function	Parameter		Friedman Test			
			MOPSO	MOSDA-NS	MOSDA-A	NSGAII
f_5	GD	Mean rank	4	2.2	2.48	**1.32(1)**
		$\rho(\gamma^2)$	0.00001(55.992)			
	DMD	Mean rank	3.8	**1(1)**	3.2	2
		$\rho(\gamma^2)$	0.00001(70.200)			
	MOS	Mean rank	4	2.4	**1(1)**	2.6
		$\rho(\gamma^2)$	0.00001(67.800)			
	HV	Mean rank	2.5	2.5	2.5	2.5
		$\rho(\gamma^2)$				
	TOC	Mean rank	**1(1)**	4	3	2
		$\rho(\gamma^2)$	0.00001(75.000)			
f_6	GD	Mean rank	2	**1(1)**	3	4
		$\rho(\gamma^2)$	0.00001(75.000)			
	DMD	Mean rank	3.56	**1(1)**	3.32	2.12
		$\rho(\gamma^2)$	0.00001(62.856)			
	MOS	Mean rank	2.64	1.76	**1.68(1)**	3.92
		$\rho(\gamma^2)$	0.00001(48.840)			
	HV	Mean rank	2.82	4	1.78	**1.4(1)**
		$\rho(\gamma^2)$	0.00001(62.207)			
	TOC	Mean rank	**1(1)**	4	2.88	2.12
		$\rho(\gamma^2)$	0.00001(71.832)			
f_7	GD	Mean rank	2.12	**1.04(1)**	2.84	4
		$\rho(\gamma^2)$	0.00001(69.624)			
	DMD	Mean rank	3.64	**1(1)**	2.6	2.76
		$\rho(\gamma^2)$	0.00001(54.408)			
	MOS	Mean rank	2.88	2.04	**1.2(1)**	3.88
		$\rho(\gamma^2)$	0.00001(59.256)			
	HV	Mean rank	2.46	4	**1.58(1)**	1.96
		$\rho(\gamma^2)$	0.00001(51.462)			
	TOC	Mean rank	**1(1)**	4	2.36	2.64
		$\rho(\gamma^2)$	0.00001(68.088)			
f_8	GD	Mean rank	2	**1(1)**	3	4
		$\rho(\gamma^2)$	0.00001(75.000)			
	DMD	Mean rank	3.88	**1.04(1)**	3.12	1.96
		$\rho(\gamma^2)$	0.00001(70.680)			
	MOS	Mean rank	3.04	2.16	**1.08(1)**	3.72
		$\rho(\gamma^2)$	0.00001(58.680)			
	HV	Mean rank	**1.82(1)**	4	2.32	1.86
		$\rho(\gamma^2)$	0.00001(47.891)			
	TOC	Mean rank	**1(1)**	3.24	3.76	2
		$\rho(\gamma^2)$	0.00001(69.528)			

Function	Parameter		Friedman Test			
			MOPSO	MOSDA-NS	MOSDA-A	NSGAII
f_9	GD	Mean rank	2.32	**1.16(1)**	3.32	3.2
		$\rho(\gamma^2)$	0.00001(44.856)			
	DMD	Mean rank	3.56	**1(1)**	2.72	2.72
		$\rho(\gamma^2)$	0.00001(52.056)			
	MOS	Mean rank	3.32	**1.32(1)**	2.48	2.88
		$\rho(\gamma^2)$	0.00001(33.144)			
	HV	Mean rank	1.9	4	2.48	**1.62(1)**
		$\rho(\gamma^2)$	0.00001(50.976)			
	TOC	Mean rank	**1(1)**	4	2	3
		$\rho(\gamma^2)$	0.00001(75.000)			
f_{10}	GD	Mean rank	2	**1.6(1)**	2.76	3.64
		$\rho(\gamma^2)$	0.00001(36.408)			
	DMD	Mean rank	**1.64(1)**	3.2	3.16	2
		$\rho(\gamma^2)$	0.00001(28.728)			
	MOS	Mean rank	2.2	2.88	3.36	**1.56(1)**
		$\rho(\gamma^2)$	0.00001(27.864)			
	HV	Mean rank	2.44	4	2	**1.56(1)**
		$\rho(\gamma^2)$	0.00001(51.218)			
	TOC	Mean rank	**1(1)**	4	2	3
		$\rho(\gamma^2)$	0.00001(75.000)			

APPENDIX C
WILCOXON TEST

		MOPSO vs MOSDA-A			MOPSO vs MOSDA -NS			NSGAII vs MOSDA-A			NSGAII vs MOSDA-NS			MOSDA-A vs MOSDA-NS		
		$R+$	$R-$	p	$R+$	$R-$	p	$R+$	$R-$	p	$R+$	$R-$	p	$R+$	$R-$	p
f_1	GD	325	0	0	323	2	0	235	90	**0.5118**	323	2	0	172	153	**0.79486**
	DMD	325	0	0	325	0	0	0	325	0	210	115	**0.20054**	325	0	0
	MOS	325	0	0	325	0	0	168	157	0.88076	325	0	0	185	140	0.54186
	HV	1	324	0	0	325	0	124.5	151.5	0.6818	0	325	0	0	325	0
	TOC	0	325	0	0	325	0	325	0	0	0	325	0	0	325	0
f_2	GD	325	0	0	325	0	0	41	284	0.00108	202	123	**0.28914**	292	33	0.0005
	DMD	109	216	**0.14986**	325	0	0	0	325	0	246	79	0.02444	325	0	0
	MOS	324	1	0	315	10	0	309	16	0.00008	129	196	**0.36812**	14	311	0.00006
	HV	0	325	0	0	325	0	108.5	167.5	**0.36812**	0	325	0	0	325	0
	TOC	286	39	0.0009	0	325	0	325	0	0	0	325	0	0	325	0
f_3	GD	325	0	0	325	0	0	1	324	0	49	276	0.00228	324	1	0
	DMD	325	0	0	325	0	0	0	325	0	325	0	0	325	0	0
	MOS	325	0	0	325	0	0	287	38	0.0008	70	255	0.01278	325	0	0
	HV	22	254	0.00042	0	325	0	192	133	**0.42952**	0	325	0	0	325	0
	TOC	0	325	0	0	325	0	325	0	0	0	325	0	0	325	0
f_4	GD	325	0	0	325	0	0	14	311	0.00006	95	230	**0.06876**	300	25	0.00022
	DMD	2	323	0	26	299	0.00024	0	325	0	173	152	**0.77948**	325	0	0
	MOS	324	1	0	322	3	0	305	16	0.00008	81	244	0.02852	23	302	0.00018
	HV	9.5	18.5	2	0	325	0	3	18	**3**	0	325	0	0	325	0
	TOC	0	325	0	0	325	0	325	0	0	0	325	0	0	325	0
f_5	GD	325	0	0	325	0	0	113	212	**0.18352**	325	0	0	229	96	**0.07346**
	DMD	294	31	0.0004	325	0	0	0	325	0	325	0	0	325	0	0
	MOS	325	0	0	325	0	0	325	0	0	189	136	**0.47777**	0	325	0
	HV	*Not valid*														
	TOC	0	325	0	0	325	0	325	0	0	0	325	0	0	325	0
f_6	GD	0	325	0	325	0	0	325	0	0	325	0	0	325	0	0
	DMD	253	72	0.01468	325	0	0	3	322	0	325	0	0	325	0	0
	MOS	318	7	0	279	46	0.00174	298	27	0.00026	325	0	0	155	170	**0.84148**

	HV	302.5	22.5	0.00016	0	325	0	73	227	0.0278	0	325	0	0	325	0
	TOC	0	325	0	0	325	0	325	0	0	0	325	0	0	325	0
f_7	GD	29	296	0.00034	324	1	0	312	13	0	325	0	0	325	0	0
	DMD	304	21	0.00014	325	0	0	220	105	**0.12114**	325	0	0	325	0	0
	MOS	309	16	0.00008	286	39	0.0009	280	45	0.00158	323	2	0	19	306	0.00012
	HV	306.5	18.5	0.0001	0	325	0	189	87	**0.12114**	0	325	0	0	325	0
	TOC	0	325	0	0	325	0	325	0	0	0	325	0	0	325	0
f_8	GD	0	325	0	325	0	0	324	1	0	325	0	0	325	0	0
	DMD	305	20	0.00012	325	0	0	0	325	0	323	2	0	325	0	0
	MOS	325	0	0	286	39	0.0009	325	0	0	323	2	0	1	324	0
	HV	119	134	**0.81034**	0	325	0	79.5	196.5	**0.07508**	0	325	0	0	325	0
	TOC	0	325	0	0	325	0	325	0	0	0	325	0	0	325	0
f_9	GD	254	71	0.0139	313	12	0	1	324	0	325	0	0	325	0	0
	DMD	277	48	0.00208	325	0	0	158	167	**0.90448**	325	0	0	325	0	0
	MOS	313	12	0	308	17	0.0001	188	137	**0.4902**	323	2	0	301	24	0.00002
	HV	70.5	229.5	0.0232	0	325	0	85.5	167.5	**0.18352**	0	325	0	0	325	0
	TOC	23	302	0.00018	0	325	0	325	0	0	0	325	0	0	325	0
f_{10}	GD	78	247	0.0232	205	120	**0.25428**	302	23	0.00018	325	0	0	320	5	0
	DMD	12	313	0	69	256	0.01174	13	312	0	95	230	0	132	193	**0.41222**
	MOS	81	244	0.02852	100	225	**0.09296**	0	325	0	15	310	0.00008	213	112	**0.17384**
	HV	168	108	**0.36282**	0	325	0	84	216	**0.05876**	0	325	0	0	325	0
	TOC	0	325	0	0	325	0	325	0	0	0	325	0	0	325	0

APPENDIX D
FRIEDMAN VS WILCOXON

Friedman				Wilcoxon		
Function	Parameter	Rank Num.	Algorithm	1 vs 2	2 vs 3	3 vs 4
f_1	GD	1	NSGAII	SIG		
		2	MOSDANS		NONSIG	
		3	MOSDAA			SIG
		4	MOPSO			
	DMD	1	MOSDANS	NONSIG		
		2	NSGAII		SIG	
		3	MOSDAA			SIG
		4	MOPSO			
	MOS	1	NSGAII	SIG		
		2	MOSDANS		NONSIG	
		3	MOSDAA			SIG
		4	MOPSO			
	HV	1	MOSDANS	SIG		
		2	MOSDAA		NONSIG	
		3	NSGAII			SIG
		4	MOPSO			
	TOC	1	MOSDAA	SIG		
		2	MOPSO		SIG	
		3	NSGAII			SIG
		4	MOSDANS			
f_2	GD	1	MOSDANS	NONSIG		
		2	NSGAII		SIG	
		3	MOSDAA			SIG
		4	MOPSO			
	DMD	1	MOSDANS	SIG		
		2	NSGAII		SIG	
		3	MOPSO			NONSIG
		4	MOSDAA			
	MOS	1	MOSDAA	SIG		
		2	NSGAII		NONSIG	
		3	MOSDANS			SIG
		4	NSGAII			
	HV	1	MOSDANS	SIG		
		2	MOSDAA		NONSIG	
		3	NSGAII			SIG
		4	MOPSO			
	TOC	1	MOSDAA	SIG		
		2	MOPSO		SIG	
		3	NSGAII			SIG
		4	MOSDANS			

Friedman				Wilcoxon		
Function	Parameter	Rank Num.	Algorithm	1 vs 2	2 vs 3	3 vs 4
f_3	GD	1	NSGAII	SIG		
		2	MOSDANS		SIG	
		3	MOSDAA			SIG
		4	MOPSO			
	DMD	1	MOSDANS	SIG		
		2	NSGAII		SIG	
		3	MOSDAA			SIG
		4	MOPSO			
	MOS	1	MOSDAA	SIG		
		2	NSGAII		SIG	
		3	MOSDANS			SIG
		4	MOPSO			
	HV	1	MOSDANS	SIG		
		2	NSGAII		NONSIG	
		3	MOSDAA			SIG
		4	MOPSO			
	TOC	1	MOPSO	SIG		
		2	MOSDAA		SIG	
		3	NSGAII			SIG
		4	MOSDANS			
f_4	GD	1	NSGAII	NONSIG		
		2	MOSDANS		SIG	
		3	MOSDAA			SIG
		4	MOPSO			
	DMD	1	MOPSO	SIG		
		2	NSGAII		NONSIG	
		3	MOSDANS			SIG
		4	MOSDAA			
	MOS	1	MOSDAA	SIG		
		2	NSGAII		SIG	
		3	MOSDANS			SIG
		4	MOPSO			
	HV	1	MOSDANS	SIG		
		2	NSGAII		NONSIG	
		3	MOSDAA			SIG
		4	MOPSO			
	TOC	1	MOPSO	SIG		
		2	NSGAII		SIG	
		3	MOSDANS			SIG
		4	MOSDAA			

Friedman				Wilcoxon		
Function	Parameter	Rank Num.	Algorithm	1 vs 2	2 vs 3	3 vs 4
f_5	GD	1	NSGAII	NONSIG		
		2	MOSDANS		NONSIG	
		3	MOSDAA			SIG
		4	MOPSO			
	DMD	1	MOSDANS	SIG		
		2	NSGAII		SIG	
		3	MOSDAA			SIG
		4	MOPSO			
	MOS	1	MOSDAA	SIG		
		2	MOSDANS		NONSIG	
		3	NSGAII			SIG
		4	MOPSO			
	HV	1				
		2				
		3				
		4				
	TOC	1	MOPSO	SIG		
		2	NSGAII		SIG	
		3	MOSDAA			SIG
		4	MOSDANS			
f_6	GD	1	MOSDANS	SIG		
		2	MOPSO		SIG	
		3	MOSDAA			SIG
		4	NSGAII			
	DMD	1	MOSDANS	SIG		
		2	NSGAII		SIG	
		3	MOSDAA			SIG
		4	MOPSO			
	MOS	1	MOSDAA	NONSIG		
		2	MOSDANS		SIG	
		3	MOPSO			SIG
		4	NSGAII			
	HV	1	MOPSO	SIG		
		2	MOSDANS		SIG	
		3	MOSDAA			SIG
		4	NSGAII			
	TOC	1	MOPSO	SIG		
		2	NSGAII		SIG	
		3	MOSDAA			SIG
		4	MOSDANS			

Friedman				Wilcoxon		
Function	Parameter	Rank Num.	Algorithm	1 vs 2	2 vs 3	3 vs 4
f_7	GD	1	MOSDA-NS	SIG		
		2	MOPSO		SIG	
		3	MOSDA-A			SIG
		4	NSGAII			
	DMD	1	MOSDA-NS	SIG		
		2	MOSDA-A		NONSIG	
		3	NSGAII			SIG
		4	MOPSO			
	MOS	1	MOSDA-A	SIG		
		2	MOSDA-NS		SIG	
		3	MOPSO			SIG
		4	NSGAII			
	HV	1	MOSDA-NS	NONSIG		
		2	MOPSO		SIG	
		3	NSGAII			SIG
		4	MOSDA-A			
	TOC	1	MOPSO	SIG		
		2	MOSDA-A		SIG	
		3	NSGAII			SIG
		4	MOSDA-NS			
f_8	GD	1	MOSDA-NS	SIG		
		2	MOPSO		SIG	
		3	MOSDAA			SIG
		4	NSGAII			
	DMD	1	MOSDANS	SIG		
		2	NSGAII		SIG	
		3	MOSDAA			SIG
		4	MOPSO			
	MOS	1	MOSDAA	SIG		
		2	MOSDANS		SIG	
		3	MOPSO			SIG
		4	NSGAII			
	HV	1	MOSDANS	SIG		
		2	MOSDAA		NONSIG	
		3	NSGAII			SIG
		4	MOPSO			
	TOC	1	MOPSO	SIG		
		2	NSGAII		SIG	
		3	MOSDANS			SIG
		4	MOSDAA			

Friedman				Wilcoxon		
Function	Parameter	Rank Num.	Algorithm	1 vs 2	2 vs 3	3 vs 4
f_9	GD	1	NSGAII	SIG		
		2	MOSDANS		SIG	
		3	MOPSO			SIG
		4	MOSDAA			
	DMD	1	MOSDANS	SIG		
		2	MOSDAA		SIG	
		3	NSGAII			SIG
		4	MOPSO			
	MOS	1	MOSDANS	SIG		
		2	MOSDAA		SIG	
		3	NSGAII			SIG
		4	MOPSO			
	HV	1	MOSDANS	SIG		
		2	MOSDAA		NONSIG	
		3	MOPSO			SIG
		4	NSGAII			
	TOC	1	MOPSO	SIG		
		2	MOSDAA		SIG	
		3	NSGAII			SIG
		4	MOSDANS			
f_{10}	GD	1	MOSDANS	NONSIG		
		2	MOPSO		SIG	
		3	MOSDAA			SIG
		4	NSGAII			
	DMD	1	MOPSO	SIG		
		2	NSGAII		SIG	
		3	MOSDAA			NONSIG
		4	MOSDANS			
	MOS	1	NSGAII	SIG		
		2	MOPSO		NONSIG	
		3	MOSDANS			NONSIG
		4	MOSDAA			
	HV	1	MOSDANS	NONSIG		
		2	MOPSO		NONSIG	
		3	MOSDAA			SIG
		4	NSGAII			
	TOC	1	MOPSO	SIG		
		2	MOSDAA		SIG	
		3	NSGAII			SIG
		4	MOSDANS			