ツアーガイド割当問題に注目した多目的最適化に対するアルゴリズムに関する研究

Doctoral Dissertation of Engineering

A Study on Algorithms for Multi-objective Optimization Focusing on Tour Guide Assignment Problem

March 2020

by

Lina Setiyani

Interdisciplinary Intelligent Systems Engineering

Graduate School of Engineering and Science

University of the Ryukyus

Doctoral Dissertation of Engineering

A Study on Algorithms for Multi-objective Optimization Focusing on Tour Guide Assignment Problem

March 2020

by

Lina Setiyani

Interdisciplinary Intelligent Systems Engineering

Graduate School of Engineering and Science

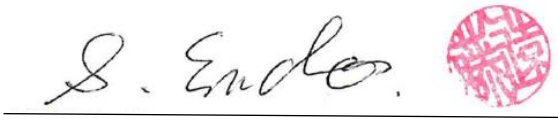
University of the Ryukyus

Supervisor: Prof. Takeo Okazaki

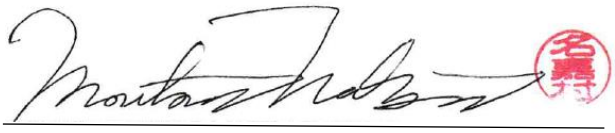
We, the undersigned, hereby, declare that we have read this dissertation and we have attended the dissertation defense and evaluation meeting. Therefore, we certify that, to the best of our knowledge this dissertation is satisfactory to the scope and quality as a dissertation for the degree of Doctoral of Engineering under Interdisciplinary Intelligent Systems Engineering, Graduate School of Engineering and Science, University of the Ryukyus.

DISSERTATION REVIEW & EVALUATION COMMITTEE MEMBERS

(Chairman) Prof. Takeo Okazaki



(Committee) Prof. Satoshi Endo



(Committee) Prof. Morikazu Nakamura

Acknowledgment

First of all, I would like to express my gratitude to my advisor Prof. Takeo Okazaki for his support, advice, and patience throughout my graduate studies. He gave me the opportunity to carry out this study and encouraged me a lot during these years. His technical and editorial advice was essential to the completion of this dissertation and he has taught me innumerable lessons and insights on the workings of academic research in general.

I would also like to say thank you to my committee members, Prof. Satoshi Endo and Prof. Morikazu Nakamura who provided invaluable feedback and suggestions to my presentation and dissertation drafts that helped me to improve my dissertation writing.

I would like to thank my friends, especially to Okazaki laboratory members who support and encourage me during my studies in the laboratory.

I would like to thank also to Rotary Scholarship Foundation, especially to my counselor Takeshi Tana Sensei and my club sponsor Naha Higashi Rotary Club for always supporting me not only in my study life but also for daily life during my stay in Okinawa.

Last, but not least, I would like to thank my husband Idam and my daughter Sachi, my parents and family for their continued support during the past few years. Their support and encouragement are essential that made this dissertation possible.

Author Publication List

1. Lina Setiyani, Takeo Okazaki, 2017, 3D Neighborhood Relationship of Cellular Genetic Algorithm for the Tour Guide Assignment Problem, IEIE Transactions on Smart Processing and Computing, vol. 6, no. 3, pp.151-157.
2. Lina Setiyani, Takeo Okazaki, 2019, NSGA-III Performance in Multi-objective Tour Guide Assignment Problem, IEIE Transactions on Smart Processing and Computing, vol. 8, no. 3, pp.219-226.

Abstract

The tour guide assignment has an important impact on a tourist service center. A well-arranged assignment could decrease the operational cost and increase the quality of service. We previously proposed and developed an optimum solution for Tour Guide Assignment Problem (TGAP) based on the cellular Genetic Algorithm (cGA) focusing on minimizing Total Guiding Time. However, in a real-world problem, the processes usually involve multiple potentially conflicting interests regarding the objectives. Considering TGAP as a multi-objective could describe the problem as close as the nature of the problem.

Multi-objective evolutionary algorithms (MOEA) are well known for solving single- and multi-objective optimization problems. In this dissertation, we conducted the optimization of the multi-objective Tour Guide Assignment Problem considering Total Guiding Time, Total Assignment Cost and Total Service Quality. Several known multi-objective evolutionary algorithms such as NSGAII, SPEAII, ε-MOEA, ε-NSGAII, PAES, PESAII, and NSGAIII have been used to solve and evaluate the problem in MOEA framework. To evaluate the quality of solutions, Hypervolume, Generational Distance and Maximum Pareto Front Error were being used as performance indicators to compute and statistically analyzed the results. We concluded that NSGAIII gave a better performance than the other algorithms in terms of solution quality and running time.

Furthermore, we applied the ε-dominance concept to maintain the diversity of the Pareto set which become the drawback in NSGAIII. Hypervolume, Generational Distance and Reference Point Convergence have been used to measure the results. We found that proposed ε-NSGAIII has a better performance compared to the original NSGAIII, ε-NSGAII, and ε-MOEA on the problem.

Also, an extended version of ε-NSGAIII named adaptive ε-NSGAIII has been proposed by introducing the adaptive ε-parameter concept to improve the algorithm. For performance analysis, we used Hypervolume, Generational Distance, Reference Point Convergence and one additional performance indicator named Spread Evaluation. As a result, we found that introducing the adaptive ε-parameter concept on ε-NSGAIII could increase the performance of the algorithm on solving Multi-objective TGAP. Finally, we tested the performance of the proposed algorithm on other real-life problem named MOTSP. As a result, adaptive ε-NSGAIII could also solve the problem with good performance.

Table of Contents

List of Figures

List of Tables

Chapter 1 Introduction

The assignment problem is a special case of linear programming problem which can be found in many life activities such as the assignment of vehicles to routes, sales to different delivery area, man/machine to jobs, teacher to a class, products to factories, contracts to bidders, machines to jobs and so on. The name is originated from the classical problem where the objective is to assign several tasks to several persons or facilities at minimum cost or maximum profit [1]. The assignment is to be made on a one-to-one basis where every task is associated with one and only one person or facility. There are several decision-making situations where assignment technique can be successfully applied [2]. Management generally makes assignments on a one-to-one basis in such a manner that the group maximizes the revenue from the sales or vehicles are deployed to various routes in such a way that the transportation cost is minimum and so on.

According to Wren basic definition of assignment is the allocation, subject to constraints, of resources to objects being placed in space-time, in such a way as to minimize the total cost of some set of the resources used [3]. The assignment problem was first investigated by G. Monge in 1784 [4] and one of the first studied combinatorial optimization problems. Up to now, there has been much research done in the field of assignment problems.

For example, the job assignment problem [5], the traffic assignment problem [6], the sailor assignment problem [7], the channel assignment problem [8], the quadratic assignment problem [9], the pin assignment problem [10], the airman assignment problem [11], the frequency assignment problem [12], the layer assignment problem [13], etc. Besides, there are also scheduling problems, such as the nurse scheduling problem [14], which also has many of the same characteristics as some of the assignment problems.

Numerous approaches have been applied by researchers to solve specific assignment problems in certain fields. Since most of the assignment problems have to deal with outside constraints on the problem and the problems typically do not have a one-to-one matching, except the quadratic assignment problem and the stable marriage problem. Many specific assignment problems including the sailor assignment problem and the nurse scheduling problem can be generalized into the constrained assignment problem [15].

Essentially, the generalization can be written to the following form:

$$min \sum_{i=1}^{N} \sum_{j=1}^{M} x_{i,j} y_{i,j} \qquad \text{(Eq. 1.1)}$$

where $x_{i,j}$ and $y_{i,j}$ are costs related to the assignment of an entity to a job. However, not all assignments may be valid. Invalid assignments are classified as hard constraints, while where assignments that are allowed but are penalized, are called soft constraints. For example, a job that requires a certain skill can eliminate people from occupying that position. Hard constraints can be viewed as Boolean types.

On the other hand, unlike the hard constraints, soft constraints can have many variations insignificance. Factors that are beneficial, but not required, to fulfill the duties of a particular job can be categorized as a soft constraint. A trait, skill, or physical quality are factors that can improve the assignment performance. However, the assignment still can be done even with the lack of these factors. So different job skills may have varying penalties associated with them. For example, a person that has the job skill specified may get a high score for that constraint, while someone who has a job skill closely related to the job may be penalized with a higher score. But someone overqualified for the job may be penalized with a much higher score.

Many researchers have contributed their work in the area of an assignment problem. For example, Ping Lee et. al. have proposed a new algorithm for the assignment problem [16]. This algorithm is simple as all the operations are performed on a *2n* x *2n* matrix. It may be an alternative to the well-known Hugarian method. Beckmann Martin and others have discussed

the assignment problem and location of economic activities [9]. Machol Robert has discussed various applications of the assignment problem [17]. Some others have developed an algorithm for the bottleneck generalized assignment problem [18]. Mc Ginnis Leon F. presented a detailed development for a computationally efficient primal-dual algorithm [19]. Using dual ascent method Murty Ishwar has suggested a procedure for solving the multi-period assignment problem having start-up cost [20]. Mathirajan M. and B.G. Raghavendra have used the assignment technique for the optimum allocation of buses to various depots [21]. Substantial saving for the BTS has been indicated by this model and resulted in a significant reduction in the dead mileage. Ross T.G. and Zoltners A. A. have described a weighted assignment model along with their applications [22]. Elshafei has suggested a method to get the solution to the problem of job assignment by using the functional equation technique of dynamic programming [23]. Subramanyam Y.V. has developed an algorithm to show the extension of the cost matrix [24]. Volgement A. has used the Core approach to solve linear and Semi-assignment problems [25].

1.1 Tour Guide Assignment Problem (TGAP)

Tour guide assignment has an important impact on a tourist service center. A well-arranged assignment could decrease operational costs and increase the quality of service. In a tour guide assignment, the process can be described as a tourist service that hires and trains tour guides to serve a group of visitors. The main purpose is to find a satisfying assignment of tour guides to groups of visitors under a set of constraints.

TGAP is based on previous work proposed by Chen, R. C. where the service center provides services to visitors [26]. The service center provides services where numbers of tour guides will guide different visitor groups. Numbers of tour guides were assigned to guide different visitor groups. Each tour guide has its own fixed guiding time, capacity and preparation time when serving a group of visitors which depends on their experience.

To maintain a good quality service, tour guides should not be assigned to guide too many visitors at a time, especially on a busy day such as holiday or weekend. To overcome this problem, when a visitor group is too large to handle, the tour center will divide a group into some smaller subgroups before being served, where each subgroup will be handled by one tour guide only (Figure 1.1).

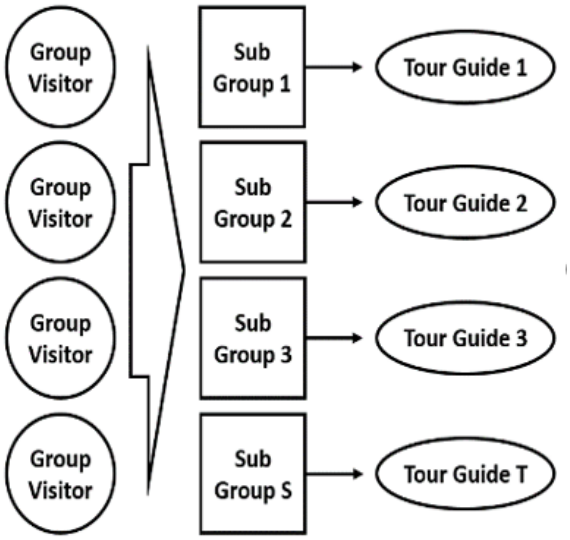


Figure 1.1 The assigning of tour guide

TGAP consists of a set of variables, a finite and discrete domain for each variable from the set and a set of constraints. The obtained solution should be able to find an assignment to all the variables such which satisfy all the constraints specified. Since the goal of TGAP is to obtain an optimal solution over a given objective function, TGAP can be specifically called as a constraint optimization problem and can be classified into a constraint satisfaction problems (CSPs)

1.2 Related Research to Solve TGAP

Previously, Chen, R. C. et. al. defined TGAP as a single objective problem. The main objective was to find a satisfying assignment of a tour guide to a group of visitors under a set of constraints [26]. By applying a genetic algorithm (GA) to the problem in optimum parameters, including crossover rates and mutation rates, they were able to obtain the best performance experimentally. Their experimental results show that GA is effective to solve the assignment problem the tour guides in the service center. In addition, they also applied a penalty function to increase the on-time service rates of visitor orders.

Zoghby et al. then used the cellular Genetic Algorithm (cGA) concept, a concept developed and introduced by Alba, C. and Dorronsoro [27], to increase the effectivity of GA in solving TGAP [28]. Aiming the same objective as was done by Chen, R. C. et. al., they were then applied the cellular genetic algorithm to assign tour guides of the service center for the tourism industry. In the optimum parameters, they were able to obtain the best performance

experimentally and revealed that the cellular genetic algorithm is more effective to solve the tour guides assignment in a service center.

Furthermore, in 2017, by applying a cGA, we showed that increasing the cellular dimension of the problem could increase the quality of the solution [29]. However, all efforts described TGAP as single-objective optimization. In reality, TGAP has more complexity. For example, at peak times, such as weekends and holidays, tourist service centers have to handle large numbers of visitors. Hiring more tour guides are needed to take care of visitors and maintain the quality of service. However, hiring more tour guides will increase operational costs. Also, TGAP has natural constraints, and the assignment should not only satisfy the tourist service center but also the customers. Considering TGAP as multi-objective could describe the problem as closely as possible to the real-world condition. Since the process of assigning a tour guide to the tourists will highly influence the problem, it is necessary to optimize it from a multi-objective point of view.

1.3 Research Aim and Objective

The main purposes of this research are:

1. Propose and develop TGAP into a multi-objective problem by considering three objectives, which are minimizing total guiding time, minimizing total assignment costs and maximizing total service quality.
2. Have a clear understanding of some well-known evolutionary algorithm on multi-objective TGAP and evaluate its quality of solutions with performance indicators to compute and statistically analyzed the results.
3. Propose and evaluate a reliable evolutionary algorithm for multi-objective TGAP.

1.4 Dissertation Structure

This dissertation is organized as follows: Chapter 1 describes Tour Guide Assignment Problem (TGAP), the difficulties and considerations of TGAP, the objectives of this research and structure of this dissertation. Chapter 2 presents a preliminary knowledge multi-objective problem and evolutionary algorithm, single-objective evolutionary algorithm, multi-objective evolutionary algorithm and performance evaluation. Chapter 3 presents a design of multi-objective TGAP including its definition, objective, restriction and metaheuristic approach to solve multi-objective TGAP. In addition, this chapter also describes the performance of some well-known evolutionary algorithms on multi-objective TGAP. Chapter 4 introduces the

epsilon concept and proposed epsilon NSGAIII method on solving multi-objective TGAP. Performance comparison to its original NSGAIII also describes in this chapter. Chapter 5 presents applied adaptive concepts on epsilon NSGAIII to increase the performance of the proposed algorithm on multi-objective TGAP. Chapter 6 resumes all work in this dissertation.

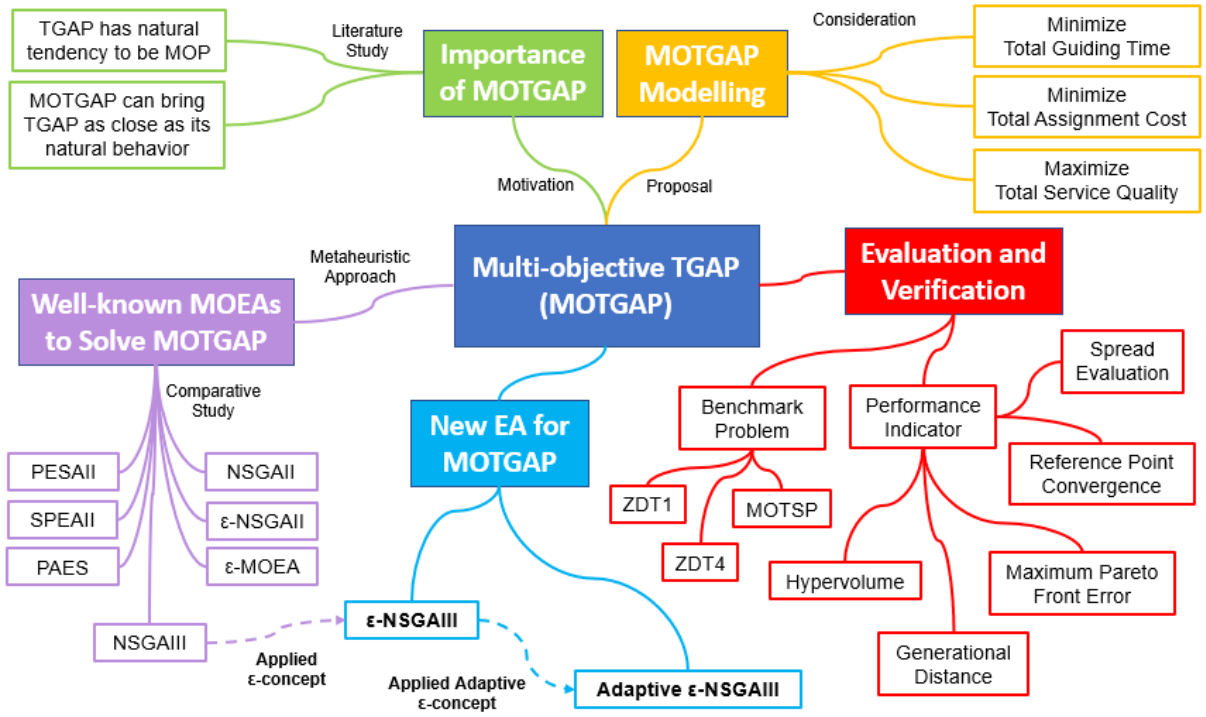


Figure 1.2 Summary of Research

Chapter 2 TGAP Modeling

2.1 TGAP

The mathematical modeling for this problem can be described as $T = \{1, 2, \dots, x\}$ for numbers of tour guides and $S = \{1, 2, \dots, y\}$ for subgroups, where $i \epsilon T, j \epsilon S$, G_{ij} can be defined as the guiding service time by guide i to be assign subgroup j [26]. The fitness function can be written as follows:

Minimize

$$F = \sum_{i=1}^{x} \sum_{j=1}^{y} \{G_{ij} V_{ij} + p_i q_i\} \quad \text{(Eq. 2.1)}$$

where

$$V_{ij} \epsilon \{0, 1\} \; \forall i \epsilon T, j \epsilon S \quad \text{(Eq. 2.2)}$$

and

$$q_i \epsilon \{0, 1\} \; \forall i \epsilon T \quad \text{(Eq. 2.3)}$$

G_{ij} is the actual guiding time, while V_{ij} will take 1 when the tour guide i is assigned to a subgroup j and 0 otherwise. In addition, p_i defines the additional preparation time where q_i takes 1 when the service sequence number of visitor subgroups in the wrong arrangement.

To set the constraints for TGAP, two main factors involved in the assignment need to be set. One is tour guides and the other is visitors. From the tour guide point of view, there are some assumptions needed to be made such as shown in Table 2.1.

Table 2.1 Tour Guide Characteristic

Parameter	Explanation
Tour Guide	The number of tour guides is known.
Working Hours	The normal working hours of a day for a guide are 8 hours.
Service Capacity	Fixed for a certain tour guide.
Preparation Time	Preparation Time for a specific tour guide is constant.

As shown in Table 2.1, the tour guide number of the service center will be set in a certain number where each tour guide has a normal 8 hours working hour in a day. Depend on their skill level, each tour guide has different service capacity and preparation time. Also, a strong restriction is applied to the tour guide wherein each assignment, each tour guide can only be assigned to one subgroup of visitors. In the next assignment, they can be assigned to other unassigned subgroups as long as they are not exceeding their normal working hours.

In addition, some data related to this experiment will also be inputted as described in Table 2.2 where service capacity and preparation time of each tour guide is different for each tour guide, depending on their skill.

Table 2.2 Important input data concerning of each guide

Parameter	Description	Value
Service capacity	High, Fair, Low	25, 20, 15
Preparation time	It depends on the tour guide. An experienced tour guide requires less preparation time.	1-2 hours

While from the visitor point of view, visitors can be divided into smaller subgroups and will be assigned to one tour guide only. Each group/subgroup of visitors has a different stay time.

Table 2.3 Visitor Characteristic

Parameter	Explanation
Visitor	It can be divided into smaller **subgroups**. Each subgroup is assigned to one tour guide only.
Stay Time	Stay Time of visitor group/subgroup: Half-day (4 hours) One day (8 hours)

2.2 Single and Multi-objective Optimization Problem

The optimization problem can be found in various fields such as science, sports, economic, engineering, planning, design, and scheduling. Even in our daily life, we can find an optimization problem without realizing it. For example, the way we manage our tasks at work, how to optimize our driving route to go to office or school, how effective we manage our money for future life, even when we think how we spent our time during vacation. The implementation of the optimization problem in life grows bigger and larger which attracts many researchers to keep finding better and more efficient techniques to solve life problems. By definition, the optimization problem refers to the process of finding one or more feasible solutions that correspond to the particular problem having one or more objectives [30].

In the beginning, many researchers have more focus on optimization problems which consider only a single objective, commonly known as Single Objective Optimization (SOO) problem. However, they realized that most real-world search and optimization problems have to consider more than one objective due to its natural behavior. The problem is, having multiple objectives in the problem also means presenting a conflict among the objectives. This situation makes the optimization problems in a multi-objective point of view more interesting and challenging to solve. This kind of optimization commonly known as Multi-objective Optimization (MOO) problems. There will be no single solution that can satisfy the multiple conflicting objectives simultaneously. For that reason, the solution to a MOO problem is a set

of trade-optimal solutions. In general, the optimization problem can be divided into several kinds, based on their variable, there are continuous and discrete optimization problems, while based on their constraints there are constraint and unconstraint optimization problems. Finally, based on their objective function, there are single and multi-objective optimization problems [31].

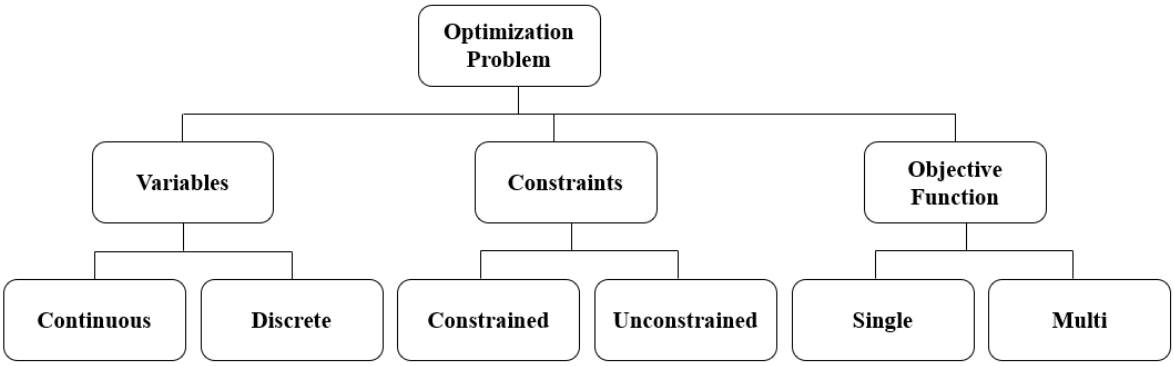


Figure 2.1 General Classification of Optimization Problem

In solving the optimization problem, there are some optimization techniques that we can use. Conventionally, there are mathematical programming and calculus method that known to be used to solve the optimization problem. However, nonconventional optimization techniques, which are known as a meta-heuristic algorithm, have attracted many researchers and have grown rapidly especially for the last decades [32]. Meta-heuristic is a general algorithmic framework that can be applied to a different optimization problem with few modifications to make them adapted to a specific problem [33]. It can be classified into several algorithms as shown in Figure 2.2 [31]. Among the meta-heuristic optimization technique, the evolutionary algorithm and swarm-based algorithm has been widely used by researchers.

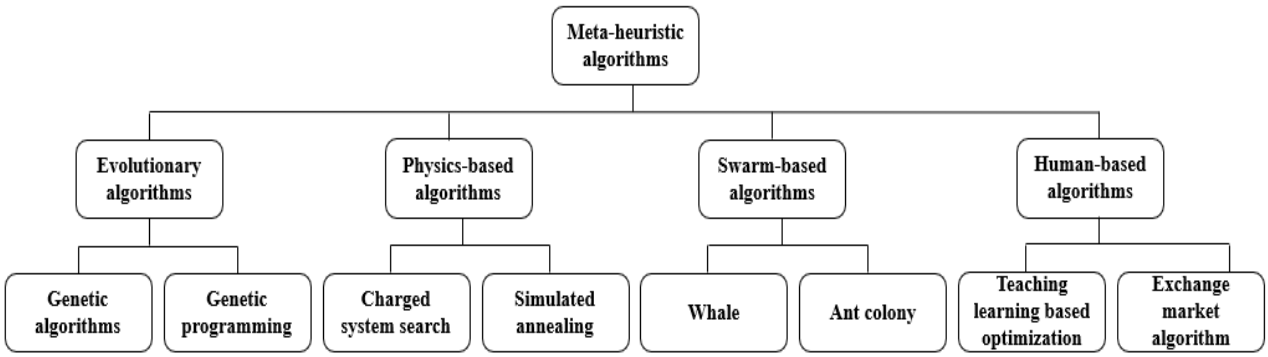


Figure 2.2 Meta-heuristic algorithms

For an SOO problem, a classical optimization method such as simulated annealing or hill climbing might be at best in finding one solution in a simulation run. However, these methods cannot solve MOO problems efficiently. Many real-world problems involve multiple measures of performance, or objectives, which should be optimized simultaneously. In certain cases, objective functions may be optimized separately from each other and insight gained concerning the best that can be achieved in each performance dimension. However, suitable solutions to the overall problem can seldom be found in this way. Optimal performance according to one objective, if such an optimum exists, often present low performance in one or more of the other objective dimensions, creating the need for a compromise to be reached.

According to Zhou A., et al. [32], in real-world problems there is no single solution that can optimize all objectives, because these problems involve multiple conflicting objectives; improving one objective may at the same time degrade another one; therefore, the best trade-off solutions are important to decision-makers. The simultaneous optimization of multiple, possibly competing, objective functions deviate from single-function optimization in that it seldom admits a single, perfect (or Utopian) solution. Instead, MOO problems tend to be characterized by a family of alternatives which must be considered equivalent in the absence of information concerning the relevance of each objective relative to the others. Multiple solutions, or multimodality, arise even in the simplest non-trivial case of two competing objectives, where both are unimodal and convex functions of the decision variables. As the number of competing objectives increases and less well-behaved objectives are considered, the problem of finding a satisfactory compromise solution rapidly becomes increasingly complex. For that reason, a set of solutions that define the best tradeoff between competing objectives have to be found.

Conventional optimization techniques, such as gradient-based and simplex-based methods, and also less conventional ones, such as simulated annealing, are difficult to extend to the true multi-objective case because they were not designed with multiple solutions in mind [33]. In practice, multi-objective problems have to be reformulated as a single objective before optimization, leading to the production of a single solution per run of the optimizer. In the SOO, the superiority of a solution over other solutions is easily determined by comparing their objective function values. However, in a MOO problem, the goodness of a solution is determined by dominance.

A multi-objective problem can be expressed in its general form mathematically as

$$\text{Minimize/Maximize } f_m(x), \qquad m = 1, 2, \ldots, M; \qquad \text{(Eq. 2.4)}$$

$$\text{subject to } g_i(x) \geq 0, \qquad j = 1, 2, \ldots, J; \qquad \text{(Eq. 2.5)}$$

$$h_k(x) = 0, \qquad k = 1, 2, \ldots, K; \qquad \text{(Eq. 2.6)}$$

$$x_i^L \leq x_i \leq x_i^U, \qquad i = 1, 2, \ldots, n; \qquad \text{(Eq. 2.7)}$$

where f_i is the *i*-th objective function and *M* is the number of objectives.

$$f(x) = [f_1(x), f_2(x), \ldots, f_m(x)]^T \qquad f(x) \in \mathbb{R}^M \qquad \text{(Eq. 2.8)}$$

A solution x is a vector of n decision variables:

$$x = [x_1, x_2, \ldots, x_n]^T \qquad \text{(Eq. 2.9)}$$

The above general problem is associated with *J* inequality and *K* equality constraints. The last set of constraints are called variable bounds, restricting each decision variable x_i to take a value within a lower $x_i^{(L)}$ and an upper $x_i^{(U)}$ bound. These variable bounds constitute the decision variable space $\Omega \in \mathbb{R}^n$, or simply the decision space. In the presence of constraints g_j and h_k, the entire decision variable space may not be feasible. The feasible region S is the set of all feasible solutions in the context of optimization. The feasible search space can be divided into 2 sets of the solutions-Pareto optimal and non-Pareto optimal set. To define Pareto optimality, first, we need to look into the concept of domination.

There are M objective functions in a MOO problem. Say, we have 2 solutions, i and j. $i < j$ implies i is better than j or i dominates j. A solution x^1 is said to dominate another solution x^2, if both the following conditions are true:

- The solution x^1 is as good as x^2 in all objectives, i.e. $f_i(x^1) \leq f_i(x^2)$, i + 1, 2, …, m. for all m = 1, 2, . . ., *M*, assuming a minimization problem.
- The solution x^1 is strictly better than x^2 in at least one objective, i.e. $f_i(x^1) < f_i(x^2)$ in at least one objective i.

Among a set of solutions (P), the non-dominated solutions (P^*) are those that are not dominated by any member of the set P. When the set P comprises the entire search space, the

resulting non-dominated set P^* is the *Pareto Optimal Set* (POS in the decision space). Pareto optimal solutions joined together as a curve form the *Pareto Optimal Front* (POF in the objective space). The front will lie in the bottom-left corner of the search space for problems where all objectives are to be minimized [34].

Given a set of solutions, the non-dominated solution set is a set of all the solutions that are not dominated by any member of the solution set. The non-dominated set of the entire feasible decision space is called the Pareto-optimal set.

The set of all Pareto-optimal decision vectors is called the Pareto-optimal, efficient, or admissible set of the problem. The corresponding set of objective vectors is called the non-dominated set.

The boundary defined by the set of all points mapped from the Pareto optimal set is called the Pareto optimal front. In practice, however, it is not unusual for these terms to be used interchangeably to describe solutions of a MOO problem. The notion of Pareto-optimality is only a first step towards the practical solution of a multi-objective problem, which usually involves the choice of a single compromise solution from the non-dominated set according to some preference information.

Evolutionary algorithms (EAs), however, have been recognized to be possibly well-suited to MOO since early in their development. Since GA was proposed for the first time in the early 1970s [35], the study of the evolutionary algorithm has emerged as a popular research field [36]. Researchers from various scientific and engineering disciplines have been digging into this field, exploring the unique power of evolutionary algorithms [37]. Multiple individuals can search for multiple solutions in parallel, eventually taking advantage of any similarities available in the family of possible solutions to the problem. The ability to handle complex problems, involving features such as discontinuities, multimodality, disjoint feasible spaces, and noisy function evaluations, reinforces the potential effectiveness of EAs in multi-objective search and optimization, which is perhaps a problem area where evolutionary computation distinguishes itself from its competitors.

EAs draw inspiration from nature. An EA starts with a randomly initialized population. The population then evolves across several generations. In each generation, fit individuals are selected to become parent individuals. They will cross-over with each other to generate new individuals, which are subsequently called offspring individuals. Randomly selected offspring individuals then undergo certain mutations. After that, the algorithm selects the optimal individuals for survival to the next generation according to the survival selection scheme designed in advance. For instance, if the algorithm is overlapping then both parent and offspring

populations will participate in the survival selection [38]. Otherwise, only the offspring population will participate in the survival selection. The selected individuals then survive to the next generation. Such a procedure is repeated again and again until a certain termination condition is met [39].

Among other EA, cGA is one of the widely-used algorithms for solving the multi-objective problem. As seen in Algorithm 1.1 [27], cGA works by first creating the population (line 2). Each individual in the population was then evaluated by their fitness function (line 3). The algorithm works on each individual in the population according to its place orderly (line 5) where the current individual interaction was limited to only his or her neighbors (line 6).

Algorithm 1.1 Pseudo-code of cGA.
1 :Proc Evolve(cga) 2 :GenerateInitialPopulation(cga.pop); 3 :Evaluation(cga.pop); 4 :While ! stop_condition() do 5 :for individual = 1 to cga.popSize do 6 :Neighbors = compute_neighbors(cga, position, (individual)); 7 :Parents = selection (neighbors); 8 :offspring = Recombine (cga.Pc, parents); 9 :offspring = Mutate(cga.Pm, offsprings); 10 :Evaluation(offspring); 11 :Replace_if_better (position(individual), auxiliary_pop, offspring); 12 :end for; 13 :cga.pop = auxiliary_pop; 14 :end while; 15 :end proc Evolve

The parents of the individual are chosen from the neighbors by using a selection technique (line 7). After the crossover operators are applied to the current individual with probabilities P_c (line 8) and mutation with probabilities P_m (line 9) the algorithm computes the fitness values of their offsprings (line 10). The best one was then inserted into new or the current population or in a new one based on replacement policy (line 11). A new population for the next generation (line 13) was then generated and keep generating until fulfilling the termination condition (line 4). The termination condition is met either by finding the optimum solution or exceeding the maximum number of calling the evaluation function or composed of both.

2.3 Evolutionary Algorithm for Single and Multi-objective Optimization

In the traditional 'single' objective approach, such as in the classic linear programming framework, one must assume that there is exactly one objective that is to be optimized subject to the absolute satisfaction of several 'constraints'. Often one of the objectives is optimized while the others are specified as constraints. For example, maximization of profit (or gross margin) or minimization of costs is considered the single most objective to be optimized.

Proponents of multiple objective approaches argue that although logically sound, the single objective approach fails to faithfully reflect the real-life decision situation for two reasons. Firstly, it assumes that the constraints that define the feasible set are so rigid that they cannot be violated. Secondly, decision-makers are usually not interested in ordering the feasible set according to just a single criterion but would rather find an optimal compromise involving several objectives. Moreover, a decision-maker or a farmer, for instance, might be involved in the diversity of occupations or activities such as farm and non-farm activities.

Optimization problems involving multiple, conflicting objectives are often approached by aggregating the objectives into a scalar function and solving the resulting single-objective optimization problem. Two major problems must be addressed when an evolutionary algorithm is applied to multi-objective optimization. First is how to accomplish fitness assignment and selection, respectively, to guide the search towards the Pareto-optimal set and second is how to maintain a diverse population to prevent premature convergence and achieve a well-distributed trade-off front. Often, different approaches are classified concerning the first issue, where one can distinguish between criterion selection, aggregation selection, and Pareto selection [40]. Methods performing criterion selection switch between the objectives during the selection phase. Each time an individual is chosen for reproduction, potentially a different objective will decide which member of the population will be copied into the mating pool. Aggregation selection is based on the traditional approaches to multi-objective optimization where the multiple objectives are combined into a parameterized single objective function. The parameters of the resulting function are systematically varied during the same run to find a set of Pareto-optimal solutions. Finally, Pareto selection makes direct use of the dominance relation. Goldberg suggested a Pareto-based fitness assignment strategy [41].

Let us consider multi-objective problem as the following form:

Minimize $\{f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x})\}$ *(Eq. 2.10)*

Subject to

$\boldsymbol{x} \in S \subset \mathbb{R}\ n$, $k \geq 2$ objectives (Eq. 2.11)

$f_i : S \rightarrow \mathbb{R}$ (Eq. 2.12)

If MOP involves an objective function f_{i} to be maximized, then we consider an equivalent objective function $-f_{\mathrm{i}}$ that is minimized. Generally, a MOP has many optimal solutions called Pareto optimal solutions with different trade-offs. A decision vector $\boldsymbol{x}* \epsilon S$ is Pareto optimal if there does not exist any other $\boldsymbol{x} \epsilon S$, such that

$f_i\ (\boldsymbol{x}) \leq f_i\ (\boldsymbol{x}*)\ \forall\ i = 1, 2, \ldots, k$ (Eq. 2.13)

and

$f_i\ (\boldsymbol{x}) < f_j\ (\boldsymbol{x}*)$ (Eq. 2.14)

for at least one index j

In simple words, a solution is Pareto optimal if no objective function value can be improved without impairing any other objective function values. An objective vector (k dimensional) is Pareto optimal if the corresponding decision vector is Pareto optimal. The set of Pareto optimal solutions in the decision space is called Pareto optimal set and their corresponding set of Pareto optimal solutions in the objective space is called Pareto optimal front. It is also common to define two different objective vectors i.e. ideal and nadir points that provide ranges of objective function values in the Pareto optimal front.

Assuming that the objective functions are bounded over the feasible region S, an ideal point can be obtained by individually minimizing each of the objective functions subject to the constraints [42].

$z_i\ ideal = \min f_i(\boldsymbol{x}),\ \boldsymbol{x} \epsilon S$, for $i = 1, 2, \ldots, k$ (Eq. 2.15)

and provides the lower bound of the objective function values of the Pareto optimal front. Meanwhile, a nadir point provides the upper bound of the objective function values of the Pareto

optimal front. However, since a nadir point is difficult to calculate it is common to approximate the nadir point by using the decision point obtained when calculating the ideal point.
Recently, methods are also proposed to find a reliable estimate of the nadir point. However, they are difficult to implement. In addition to the ideal and nadir points, it is also common to define an additional objective vector called a utopian objective vector, which can be calculated as

$$z_i\ utopia = z_i\ ideal - \varepsilon_i \ \text{, for } i = 1, 2, \ldots, k \text{ and } e_i > 0 \qquad \text{(Eq. 2.16)}$$

The importance of the utopian point is limited to account for the numerical problems that arise when ideal and nadir points are close to each other. As mentioned before, there exist several Pareto optimal solutions to a Multi-objective Optimization Problem and a decision-maker to express preference information.

A common way is to express the preference information in terms of desirable values of objective functions ($\bar{z_i}$, $i = 1, 2, \ldots, k$), often termed as a reference point.

2.4 Metaheuristic approach to solve MOO Problems

In multi-objective optimization, as there are more than two conflicting objectives to be solved, the main challenge is to determine a solution that can satisfy all objectives in TGAP. However, since it is difficult to satisfy all objectives at the same time, a set of Pareto optimal solutions that can be used to find the best compromise has to be determined. The multi-objective optimization evolutionary algorithm (MOEA) is a well-known method that can be used as an approach to solving a multi-objective problem. Among the algorithms, the Non-dominated Sorting Genetic Algorithm II (NSGAII) is the most widely used [43]. ε-NSGAII was then introduced to improve the efficiency and reliability of the original NSGAII [44]. However, it was found that NSGAII has a low number of uniform diversities. Besides, there is no balance in the exploration and exploitation of the solutions due to the absence of a lateral diversity preserving operator in NSGAII.

A powerful technique named NSGAIII was then introduced by Deb and Jain [45] to overcome the disadvantage in NSGAII. In 2016, Rajnikant et al. showed that NSGAIII could improve the solution obtained in a non-linear problem with a better result compared to other algorithms [46]. In this study, PAES [47] and PESAII [48] are used as baselines for comparing the algorithms. Also, SPEAII used for comparison. SPEAII is an old popular baseline algorithm

that uses a fine-grained fitness assignment strategy for ranking solutions, a density estimation technique, and an enhanced archive truncation method [49]. The effectiveness and reliability of the MOEAs are evaluated and compared using different performance indicators from the literature.

2.4.1 NSGAII

The NSGAII is a second-generation classic example of a Pareto-based MOEA introduced by Deb et al. that incorporates improvements comparing to the original algorithm NSGA. NSGAII is historically used as a benchmarking MOEA and together with its ancestor it is the first MOEA to use Pareto dominance relation to search for the Pareto front in a single run [43].

There are three significant differences between the original NSGA and the improved one. First, NSGAII is more efficient having less computational complexity compared to NSGA. Second, NSGAII eliminates the need for parameter sharing. Third, NSGAII uses a selection operator that can combine the parent and offspring populations and select the best N solutions, considering the fitness and solutions ‘spreading, to capture a more extended Pareto surface.

Also, NSGAII supports both real and binary decision variables for output representation which also makes it suitable for our work. Zitzler et al. and Deb et al. have documented that NSGAII can perform at least as well as or better than other second-generation MOEAs when solving difficult multi-objective optimization problems [50].

The NSGAII algorithm has three prominent characteristics: (i) fast non-dominated sorting approach, (ii) fast crowded-distance estimation, (iii) simple crowded-comparison method (niche comparison) [43] which are described as in algorithm 2.1.

Algorithm 2.1 Pseudo-code of NSGAII.

1: **procedure** NSGAII (N,N_A)
2: $t \leftarrow 0$
3: $P_t \leftarrow$ **new_population** (N)
4: $Q_t \leftarrow \emptyset$
5: $A \leftarrow$ **non_dominated** (P_t)
6: **while** not stop criterion **do**
7: $R_t \leftarrow P_t \cup Q_t$
8: $\mathcal{F} \leftarrow$ **fast_non_dominated_sorting** (R_t)
9: $P_{t+1} \leftarrow \emptyset$
10: $i \leftarrow 1$
11: **while** $|P_{t+1}| + |\mathcal{F}_i| \leq N$ **do**
12: $C_i \leftarrow$ **crowding_distance_assignment** ($\mathcal{F}_i$)
13: $P_{t+1} \leftarrow P_t \cup \mathcal{F}_i$
14: $i \leftarrow i + 1$
15: **end while**
16: $\mathcal{F}_i \leftarrow$ **sort**$(\mathcal{F}_i, C_i, 'descending')$
17: $P_{t+1} \leftarrow P_{t+1} \cup \mathcal{F}_i[1: (N - |P_{t+1}|)]$ ⊳ fill P_{t+1} With the $N - |P_{t+1}|)]$ ⊳ fill P_{t+1} less crowded individuals of $\mathcal{F}_i$
18: $Q_{t+1} \leftarrow$ **selection**(P_{t+1}, N)
19: $Q_{t+1} \leftarrow$ **crossover**(Q_{t+1})
20: $Q_{t+1} \leftarrow$ **mutation**(Q_{t+1})
21: $t \leftarrow t + 1$
22: $A \leftarrow$ **non_dominated**$(A \cup Q_t)$
23: **end while**
24: **end procedure**

2.4.2 SPEAII

The SPEAII is an extended version of the SPEA that exploits the improved schema for fitness assignment to the solutions, considering a faster kth nearest neighborhood to provide more diversity among solutions [49]. The classic SPEA uses a clustering approach instead. The SPEAII is one of the popular MOEAs introduced shortly after NSGAII. The most significant difference between NSGAII and SPEAII is that the NSGAII uses a pairwise Pareto dominance comparator to measure the fitness of solutions. In contrast, the SPEAII assigns a single fitness value to each solution based on the number of competing solutions they dominate. Besides the similarities and popularity of the two algorithms, NSGAII and SPEAII are both considered as historical benchmarking algorithms compared to more recent MOEAs.

Algorithm 2.2 Pseudo-code of SPEAII
1: Set $k = 0$ 2: Initialize P_0 and set $A_0 = \emptyset$ 3: while $k < G$ do 4: Calculate fitness for A_k and P_k 5: Copy all non-dominated solutions from $P_t \cup A_k$ to A_{k+1} 6: Use the truncation operator to remove elements from A_{t+1}, if its capacity has been extended 7: Use dominated individuals from A_k to fill A_{k+1}, if its capacity has not been exceeded. 8: Perform selection on A_{k+1} to fill the mating pool 9: Apply recombination and mutation operators to obtain P_{k+1} 10: Set k = k + 1 11: end 12: return A_k

2.4.3 ε-MOEA

The ε-MOEA is a steady-state and modern MOEA that uses the ε-dominance parameterization allowing the user to define desired objective precision and find more diverse solutions. ε-dominance relation and ε-dominance archive introduced by Laumanns et al. to resolve the problem of deterioration of the approximation set [51].

Definition 1 (Dominance relation)

Let $f, g \in \mathbb{R}^m$. Then f is said to dominate g, denoted as $f \succ g$,

$$\forall_i \in \{1, 2, \ldots, m\} : f_i \geq g_i \quad \text{(Eq. 2.17)}$$

$$\exists_j \in \{1, 2, \ldots, m\} : f_j \geq g_j \quad \text{(Eq. 2.18)}$$

Based on the concept of dominance, the Pareto set can be defined as follows.

Definition 2 (Pareto set)

Let $F \subseteq \mathbb{R}^m$ be a set of vectors. Then the Pareto set $F^*$of F is defined as follows: F^* contains all vectors $g \in F$ which are not dominated by any vector $f \in F$, i.e.

$$F^* := \{g \in F | \nexists f \in F : f \succ g\} \quad \text{(Eq. 2.19)}$$

Moreover, for a given set F, the set F^* is unique. Therefore, we have $P^*(F) = \{F^*\}$. For many sets F, the Pareto set F^* is of substantial size. Thus, the numerical determination of F^* is prohibitive, and F^* as a result of the optimization is questionable. Moreover, it is not clear at all what a decision-maker can do with such a large result of an optimization run. What would be more desirable is an approximation of F^* which approximately dominates all elements of F and is of polynomially bounded size. This set can then be used by a decision-maker to determine

interesting regions of the decision and objective space which can be explored in further optimization runs.

Definition 3 (ε-Dominance)

Let $f, g \in \mathbb{R}^{+m}$. Then f is said to ε-dominate g for some $\varepsilon > 0$, denoted as $f \succ_\varepsilon g$, iff for all $i \in \{1, 2, \ldots, m\}$

$$(1 + \varepsilon) \cdot f_i \geq g_i \qquad \text{(Eq. 2.20)}$$

Definition 4 (ε-approximate Pareto set)

Let $F \subseteq \mathbb{R}^{+m}$ be a set of vectors and $\varepsilon > 0$. Then a set F_ε is called an ε-approximate Pareto set of F, if any vector $g \in F$ is ε-dominated by at least one vector f $\in F_\varepsilon$, i.e.

$$\forall_g \in F : \exists f \in F_\varepsilon \text{ such that } f \succ g \qquad \text{(Eq. 2.21)}$$

The set of all ε-approximate Pareto sets of F is denoted as $P \succ_\varepsilon (F)$.

The set F_ε is not unique. Many different concepts for ε-efficiency and the corresponding Pareto set approximations exist in the operations research literature, a survey is given by Helbig and Pateva (1994).

Definition 5 (ε-Pareto set)

Let $F \subseteq \mathbb{R}^{+m}$ be a set of vectors and $\varepsilon > 0$. Then a set $F_\varepsilon^* \subseteq F$ is called an ε-Pareto set of F if

1.F_ε^* is an ε-approximate Pareto set of F, i.e. $F_\varepsilon^* \in P_\varepsilon (F)$, and

2. F_ε^* contains Pareto points of F only, i.e. $F_\varepsilon^* \subseteq F^*$.

The set of all ε-Pareto sets of F is denoted as $P_\varepsilon^* (F)$.

Since finding the Pareto set of an arbitrary set F is usually not practical because of its size, one needs to be less ambitious in general. Therefore, the ε-approximate Pareto set is a practical solution concept as it not only represents all vectors F but also consists of a smaller number of elements. Of course, an ε-Pareto set is more attractive as it consists of Pareto vectors only.

In other words, this property maintains a fixed-size population. Also, a dynamic size for the ε-dominance archive will lead to an algorithm that can grow and shrink as needed, to provide an approximation representation of the Pareto front.

In addition, the ε-MOEA exploits efficient parent and archive update strategies (Algorithm 2.3). Together these properties enabled the algorithm to achieve well-distributed solution sets in an appropriate time and guarantee both diversity and convergence.

Algorithm 2.3 Pseudo-code of ε-MOEA.

```
1: procedure ε-MOEA
2: Set t = 0
3: Initialize new population P_t
4: Put ε non-dominated solutions from P_t into an archive population E_t
5:      while not stop criterion do
6:       Choose one individual from E_t, and one from P_t
7:       Mate the individual and produce an offspring, Q_t
8:        Create a special array B for Q_t, which consists of abbreviated versions of the
          objective values from Q_t
9:        Insert Q_t into the archive population E
10: Domination check by using the B array instead of actual objective values
11:      if Q_t dominates a member of the archive,
12.           then replace the member with Q_t
13:         end while
14: end procedure
```

Deb et al. indicated that ε-MOEA performs as well as or better than other second-generation MOEAs in terms of convergence, diversity and computational time when solving difficult multi-objective optimization problems.

2.4.4 ε-NSGAII

The ε-NSGAII is an improved extension of NSGAII, featuring four highlighted specifications that makes it more efficient, reliable and easy-to-use compared to the original NSGAII [44]. ε-dominance archiving, dynamic population sizing, randomized restart, and automatic termination are the most significant changes in the ε-NSGAII. The primary idea behind ε-NSGAII is to minimize the traditional parameterization and include quality search properties for specific problems.

The ε-NSGAII uses a series of connected processes where small populations are first initially exploited to precondition search and automatically adapt population size commensurate with problem difficulty. As the search progresses, the population size is automatically adapted based on the number of ε-non-dominated solutions that the algorithm has found. ε-non-dominated solutions found after each generation are stored in an archive and subsequently used to direct the search using a 25% injection scheme. In the injection scheme, 25% of the subsequent population will be composed of the ε-non-dominated archive solutions and the other 75% will be generated randomly (Algorithm 2.4). This assists the search in two

ways: (i) by directing the search using previously evolved solutions and (ii) by adding new solutions to encourage the exploration of additional regions of the search space. This injection scheme bounds the population size to four times the number of solutions that exist at the user-specified ε resolution.

Algorithm 2.4 Pseudo-code of ε-NSGAII.

1: **procedure** ε-NSGAII (N,N_A)
2: Set $t = 0$
3: Initialize new population P_t (N)
4: Store ε non-dominated solutions from P_t into an archive population E_t
5: **for** i = 1 *to generation number (t)* **do**
6: Create population R_t compose of 25% ε-non-dominated archive solutions and 75% generated random population.
7: Fast non dominated sorting $\mathcal{F}$ on population R_t
8: **while** the new population P_{t+1} is being constructed from population R_t **do**
9: Crowding distance tournament selection
10: **end while**
11: Sort population P_t and divide it into a number of non-dominated fronts (F_1, F_2, ... F_l)
12: Store the best ε non-dominated solutions
13: Apply selection, crossover, and mutation to produce new offspring
Repeat until the specified number of generations is accomplished
14: **end while**
15: **end procedure**

Furthermore, ε-NSGAII has better computation compared to NSGAII because ε-NSGAII eliminates random seed analysis and also eliminates unnecessary runs by terminating search according to user-defined problem-specific precision goals, giving better computational time.

2.4.5 PAES

Knowles and Corne suggested a simple elitist MOEA using a single parent, single child (1+ 1)-evolutionary algorithm called Pareto-Archived Evolution Strategy (PAES) [47]. If a new solution is not dominated by any archive member it is included in the archive, deleting, in turn, all members that it dominates. If the archive would exceed its maximum size, the acceptance of new solutions is decided by a histogram-like density measure over a hyper-grid division of the objective space. This archiving strategy is similar to the one proposed by Kursawe, who already used an adaptive distance measure to maintain a good spread of non-dominated solutions in a fixed-size archive [52].

Algorithm 2.5 Pseudo-code of PAES

1: Set $t := 0$;
2: Initialize(c); /*Generate initial random solution*/
3: Evaluate(c); /*Evaluate of initial solution*/
4: AddToArchive(c); /*Add c to archive*/
5: while(not(Termination()))
/*Start Immune phase*/
6: (c_1^{clo}, c_2^{clo} := Cloning(c); /*Clonal expansion phase*/
7: (c_1^{hyp}, c_2^{hyp} := Hypermutation(c_1^{clo}, c_2^{clo}); /*Affinity maturation phase*/
8: Evaluation(c_1^{hyp}, c_2^{hyp}); /*Evaluation phase*/
9: if(c_1^{hyp} $dominates$ c_2^{hyp}) m := c_1^{hyp};
10: else if(c_2^{hyp} $dominates$ c_1^{hyp}) m := c_2^{hyp};
11: else m := Best(c_1^{hyp}, c_2^{hyp});
12: AddToArchive(WorstBest(c_1^{hyp}, c_2^{hyp})); /*max E_{charmm} selection*/
/*End Immune phase*/
/*Start (1+1)-PAES*/
13: if (c dominates m) discard m;
14: else if (m dominates c)
15: AddToArchive(m);
16: $c := m$;
17: else if(m is dominated by any member of the archive) discard m;
18: else test($c, m, archive_{size}, depth$);
19: $t := t + 1$;
20: end while

The PAES uses a simple local search evolution strategy combined with a historical archive that stores some of the non-dominated solutions from previous searches. The PAES uses this archive as a reference set that is compared with each mutated individual. Furthermore, by exploiting this feature, the PAES can precisely estimate the quality of new candidate solutions and is capable to find high-quality solutions in terms of diversity.

2.4.6 PESAII

The PESAII is the extended version of PESA first introduced by Corne et al. [48]. In PESAII, the selection is region-based and the selection of subject is based on a hyper box (a two-dimensional depiction of an N-dimensional box). This revision reduces the computational cost associated with Pareto ranking in PESAII. Knowles et al. [48] document that PESAII outperforms random search in terms of coverage when the number of Pareto optimal solutions is less than approximately one-quarter of the search space.

Algorithm 2.6 Pseudo-code of PESAII
1: Create empty population: Internal Population (IP) and External Population (EP); 2: Initialize and evaluate IP 3: Update EP 4: **while** *max no of generations not reacted* **do** 5: Select individuals from EP 6: Apply recombination and mutation 7: Compute fitness values of children 8: Update EP according to the crowding strategy 9: **end while** 10: Return EP

2.4.7 NSGAIII

NSGAIII is the modified version of a well-known multi-objective evolutionary algorithm, NSGAII. The modification was made to overcome NSGAII limitation in solving many objectives problem. Basically, the main components of NSGAIII are similar to NSGAII, however, significant changes were made in its selection mechanism [45]. Same as NSGAII, soon after maintaining the convergence pressure, which drives the solution candidate into Pareto front, NSGAIII uses the same dominance-based selection as NSGAII, which called fast non-dominated sorting. Based on the ranking, when the new population P_{t+1} is constructed from the combined population $R_t = P_t \cup Q_t$ and if the size of S_t is greater than the population size N, then the best members in F_l with the largest crowding distance values will be selected. Let S_t be the population selected so far (including the last non-dominated front F_l). After that, different from NSGAII, which use niching counts in the selection process, the best members from the last non-dominated front F_l in NSGAIII are selected based on the supplied reference points (see algorithm 2.7). The distance of the candidate solutions will be calculated and compared with the closest reference line specified by each weight vector.

The structured reference points were create based on the procedure proposed by Das and Dennis [53]. According to them, the reference points places on a normalized hyper-plane – a $(M-1)$ dimensional unit simplex – which is equally inclined to all objective axes and has an intercept of one on each axis. If p divisions are considered along each objective, the total number of reference points (H) in an M-objective problem is given by:

$$H = \binom{M + p - 1}{p} \qquad \text{(Eq. 2.22)}$$

Each reference point will be associated with each population members and since the created reference points are widely distributed on the entire normalized hyperplane, the obtained solutions are also likely to be widely distributed on or near the Pareto-optimal front.

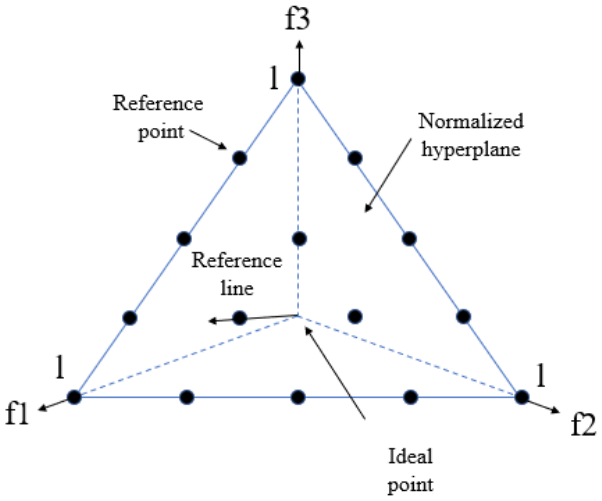


Figure 2.3 Reference Point Distribution in NSAIII

According to Deb and Jain [45], in the case of a user-supplied set of preferred reference points, ideally, the user can mark H points on the normalized hyper-plane or indicate any H, M-dimensional vectors for the purpose. This normalization step is really significant in the feature of NSGAIII. In a multi-objective problem, each objective function will naturally have different ranges of values. The normalization process will make these values to be normalized into the same range of values. The algorithm process will follow the step shown in Algorithm 2.7.

Algorithm 2.7 Pseudo code of NSGA-III.
1. Define a set of reference points 2. Initialize population (P_0) 3. Generate random population (size N_{pop}) 4. **for** i = 1 *to generation number (t)* **do** 5. Produce new offspring population Q_t through mutations and recombination 6. Combine the population ($R_t = P_t \cup Q_t$) 7. Sort population P_t and divide it into a number of non-dominated fronts (F_1, F_2, ... F_l) 8. Continue the process until F_l is reached 9. Normalize the population 10. Associate with reference points 11. Apply the niche preservation mechanism 12. Keep the obtained solutions for the next generation 13. end **for**

The algorithm is likely to find near Pareto-optimal solutions corresponding to the supplied reference points, thereby allowing this method to be used more for a combined application of decision-making and many-objective optimization.

So far, NSGAIII has a comparatively better capability in handling many objectives in some multi-objective problems.

2.5 Performance Evaluation

In the single optimization, the algorithm performance can be evaluated by the difference between $f(x)$ and function optimal value. However, the method cannot be adopted in MOPs. It is difficult to measure the performance of a multi-objective optimization algorithm by a single measurement. To solve the problem, many criteria are proposed to evaluate the performance of MOEAs. Furthermore, good performance can be defined in different ways from one algorithm to another. Since the MOEAs are initialized with randomly generated populations and the operators are probabilistic, it is crucial to evaluate the performance of the algorithms. An ideal or near-ideal algorithm is able to satisfy the three multi-objective optimization goals: proximity, diversity, and consistency. Proximity or convergence expresses how close the approximation solution set is to the solutions in the reference set. Diversity indicates the trade-offs among competing and conflicting objectives. Consistency determines whether the algorithm can discover all regions of the ideal solution space.

In multi-objective optimization, the quality of obtained non-dominated solution set is different from each algorithm however, quantitative comparison of the performance of different algorithms is an important issue. To solve this issue, a performance evaluation of each algorithm used in multi-objective optimization needs to be done. Since evolutionary algorithms perform well in finding multi-objective solutions and the outcome is usually an approximation of the Pareto-optimal front, denoted as an approximation of non-dominated solution set, performance evaluation can be done by measuring the quality of the solution set. There are two ways to judge the quality of non-dominated solution sets [53]. First, after the decision space is projected into the target space, it is dependent on the distance between the retrieved and actual Pareto Front (PF) surface. The smaller the distance, the better the convergence. Second, it depends on the distribution of the PF surface. The more uniform the distribution, the better the PF.

In an evaluation process, benchmark functions are one of the main factors. Some problems have been extensively applied by researchers to test and prove the applicability of

EMOAs. As already discussed in Section 2.1, in the development of EMOAs they have been improved in terms of less complexity. While in contrast, benchmark functions have been improved in terms of high complexity. These developing, controllable, yet challenging test problems, have been modified to test the algorithms in terms of investigating the problem difficulties; identifying such problems, with different features and characteristics, helped developers and researchers to obtain a better insight on the performance of the algorithms and find more efficient methods. Such EMOA test problems may cause the difficulty of the algorithms in converging to the true Pareto-optimal front.

Despite all the developments, there is no systematic study to speculate what problem features may cause an EMOA to face difficulties, but in this regard, there are two common types of test problems have been designed [54]. First is test problems with a specific feature that challenges a specific aspect of EMOAs. This kind of test problems that involve a particular feature that is known to cause difficulty in the optimization process, for example, multimodality and deception. Investigating the different problem features separately provides researchers with the ability to design the kind of problems in which a certain algorithm is or is not well suited. The second type is test problems having different features in order to challenge the methodologies from different aspects.

The most popular test problems referred to as standard benchmark functions are very useful in various research activities on EMOAs, such as: evaluating the performance of a new algorithm, comparing different algorithms and showing how an algorithm works. Benchmark functions are one of the main factors in the evaluation process. Among several proposed test functions, the most cited test function suits are the bi-objective Zitzler-Deb-Thiele benchmark suite known as ZDT and the scalable Deb-Thiele-Laumans-Zitzler benchmark suite known as DTLZ [27]. Although they do not reflect necessarily the main features of the real-world problems, they can determine the approximate of the Pareto-optimal front analytically with their special characteristics and features such as non-convexity, multimodality, non-uniformity of the search space, and discontinuity. In this research, we applied ZDT4 and ZDT1 test functions to evaluate the behavior and performance of the algorithm proposed in this study.

- **ZDT1**

$$f_1(x) = x_1$$
$$f_2(x) = g(x)\left[1 - \sqrt{\frac{f_1(x)}{g(x)}}\right]$$
$$g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i, x_i \in [0,1] \forall i \in \{1, \dots, n\}, n = 30 \qquad \text{(Eq. 2.23)}$$

- **ZDT4**

$$f_1(x) = x_1,$$
$$f_2(x) = g(x)\left[1 - \sqrt{\frac{f_1(x)}{g(x)}} - \frac{f_1(x)}{g\ (x)} Sin\big(10\pi f_1(x)\big)\right]$$
$$g(x) = 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i\,, x_i \in [0,1] \forall i \in \{1, \dots, n\}, n = 30 \qquad \text{(Eq. 2.24)}$$

EMOAs may have some difficulties in achieving their binary trade-off tasks (guide the search towards the global Pareto optimal region, and maintain the population diversity in the current non-dominated front), as some of them follows: the difficulty in (i) converging to the optimal Pareto front because of multimodality, deception, isolated optimum, and collateral noise, (ii) maintaining diverse Pareto optimal solutions because of convexity or non-convexity in the Pareto-optimal front, discontinuity in the Pareto-optimal front, and non-uniform distribution of solutions in the Pareto-optimal front, (iii) constraints, when hard constraints in a multi-objective problem may cause further difficulties in both previously mentioned aspects.

The benchmark functions contain important characteristics that make them highly complex problems, even more complicated than the optimization problems in reality, thus, if an EMOA can show promising results in solving the standard test functions, it can be guaranteed in some cases that the EMOA is able to solve the real-world optimization problems as well [55, 56, 57]. One of the most well-known real-world optimization problems is the Travelling Salesman Problem (TSP).

TSP is an optimization problem that has a goal to find the shortest traveling path through the given number of cities [58]. TSP describes as a traveler who has to travel through all the given number of cities N exactly once in a certain distance or time to travel between the cities and return to the same city from where he started by considering the cost of the path as minimum as possible. The cost of the path determined as the path length or travel time. The TSP mathematical model can be described as follows:

$$\min Z = \sum_{i=1}^{N}\sum_{j=1}^{N} X_{ij} C_{ij} \qquad \text{(Eq. 2.25)}$$

$$\text{subject to} \quad \sum_{i=1}^{N} X_{ij} = 1 \qquad i = 1, 2, \dots, N \qquad \text{(Eq. 2.26)}$$

$$\sum_{j=1}^{N} X_{ij} = 1 \qquad j = 1, 2, \dots, N \qquad \text{(Eq. 2.27)}$$

Where C_{ij} is the cost of traveling from i-th city to j-th city and X_{ij}= 1 if the salesman travels from city-i to city-j, otherwise X_{ij} = 0.

It is known that TSP is a naturally NP-hard problem. In real life, it is natural that TSP is not just considering one objective, i.e. the minimum distance. They also have to consider time, cost and risk. However, when the number of objectives is increasing, it is difficult to provide an optimal solution in existing solutions at the appropriate time. In addition, the solution found by optimizing the distance to travel to the destination may not always give the best solutions. For this reason, researchers started to consider TSP with more than one objective in finding the optimum solution. As a result, a better optimum solution was obtained when more objectives functions are considered, compared to solutions given by a single objective function. By considering these factors, TSP will become a Multi-objective Traveling Salesman Problem (MOTSP) [59].

In MOTSP, the aim is to simultaneously optimize several conflicting objectives mentioned before, such as the shortest traveling distance, minimum time, minimum cost, and/or lowest risk. As in other multi-objective optimization, there will be no single point is considered as an optimal solution. The obtained optimal solution will be a set of trade-off solutions among conflicting objectives. Generally, the simplest MOTSP can be formulated as a multi-objective model with two objective functions. The first objective function considers the minimization of the distance traveled by the salesman, while the second objective function considers the working time of the salesman. The MOTSP can be modeled as follows:

$$\min Z_1 = \sum_{i=1}^{N} \sum_{j=1}^{N} X_{ij} C_{ij} \qquad \text{(Eq. 2.28)}$$

$$\min Z_2 = \sum_{i=1}^{N} \sum_{j=1}^{N} X_{ij} T_{ij} \qquad \text{(Eq. 2.29)}$$

$$\text{subject to} \quad \sum_{i=1}^{N} X_{ij} = 1 \qquad i = 1,2, \ldots, N \qquad \text{(Eq. 2.30)}$$

$$\sum_{j=1}^{N} X_{ij} = 1 \qquad j = 1,2, \ldots, N \qquad \text{(Eq. 2.31)}$$

where C_{ij} is the cost for traveling from i-th city to j-th city. T_{ij} is the travel cost from i-th city to j-th city and X_{ij} = 1 if the salesman travels from city-i to city-j, otherwise X_{ij} = 0. Both the objectives are optimized to find the best optimum solutions.

2.5.1 Hypervolume

The hypervolume indicator can be formulated as:

$$HV \equiv \{\cup_i area_i | vec_i \subset P_{known}\} \quad \text{(Eq. 2.32)}$$

where vec_i is a non-dominated solution in P_{known}, and $area_1$ is the area between the origin and solution vec_i. As an indicator, the median values for each algorithm represent the ability of the algorithm to generate an approximation of the value near the set's value [60].

2.5.2 Generational Distance

Generational distance can be measured to check the ability of the algorithms to generate approximation sets that are close to the reference set:

$$GD \equiv \frac{\left(\sum_{i=1}^{n} d_i^p\right)^{\frac{1}{p}}}{|P_{known}|} \quad \text{(Eq. 2.33)}$$

where $|P_{known}|$ is the number of solutions in P_{known} and d_i^p, with $p = 2$, is the Euclidean phenotypic distance between each solution i in P_{known} towards the closest solution in P_{true}. The generational distance indicator measures the proximity of the approximation set to the reference set generated by the algorithms [60].

2.5.3 Maximum Pareto Front Error

The maximum Pareto front error represents the maximum error band and indicates the ability of the approximation set to be generated, compared to the reference set. It can be described by the following equation:

$$ME = max_j \left\{ \left\{ min_j \left(\sum_{k=1}^{m} \left| f_k^i(x) - f_k^j(x) \right|^p \right)^{\frac{1}{p}} \right\} \right\} \quad \text{(Eq. 2.34)}$$

where i and j are the index for solutions in P_{known} and P_{true}, respectively.

The maximum error band measures the largest minimum distance between each solution in the approximation set and the closest corresponding solution in the reference set. The maximum Pareto front error determines a maximum error band and indicates how well the approximation set is generated compared to the reference set. The maximum error band measures the largest minimum distance between each solution in the approximation set and the closest corresponding solution in the reference set. By definition, a small value for this indicator is more desirable for the algorithm [60].

2.5.4 Reference Point Convergence

Reference point convergence indicator quantifies the ability of convergence of an algorithm towards a reference point or a set of reference points, instead of a reference Pareto optimal set. This indicator can be applied specifically for the performance evaluation of the reference point-based algorithms. In other words, the average improvement in reference point distance of the last n populations is used as the progress indicator. The generation-based optimization progress was measured by calculating the average reference point d distance of the population, on average for the last n populations.

$$RPC_n = \frac{d_1+d_2+d_3+\cdots+d_n}{n} \quad \text{(Eq. 2.35)}$$

2.5.5 Spread Evaluation

This spread evaluation technique measured the distribution of solutions in the obtained Pareto front over the non-dominated region. It measures the distances between solutions of the obtained Pareto-front and the extent of solutions into account in order to quantify the performance of an algorithm in terms of the diversity of the obtained solutions. The smaller is the value of the Spread, the better is the performance. An ideal set of obtained non-dominated solutions will have Spread value = 0. The measure calculation will follow the formula displays on the equation below.

$$\Delta = \frac{\sum_{m=1}^{M} d_m^e + \sum_{i=1}^{|Q|} |d_i - \bar{d}|}{\sum_{m=1}^{M} d_m^e + |Q|\bar{d}} \quad \text{(Eq. 2.36)}$$

where Q is the set of obtained Pareto-front solutions, d_i is the distances between neighboring solutions, $\bar{d}$ is the mean value of the distances, and d_m^e is the distance between the extreme solutions of the global Pareto front (P^*) and the obtained Pareto front (Q) corresponding to the m^{-th}objective function [61].

Chapter 3 Multi-objective TGAP

3.1 Introduction

As a single objective, TGAP objective was previously only considered to find a satisfying assignment of a tour guide to a group of visitors based on their guiding time, where faster guiding time will be preferred. Searching for the solution will be simply directly focusing on minimum guiding time. Targeting the same single objective TGAP, we were also able to solve TGAP in our previous research by enhancing the neighborhood dimension in cGA.

However, looking at the nature of the problem, we realized that the main difficulties in solving TGAP, as most of the other assignment problem, is highly constrained nature and the environmental conditions are different for each service center such as working hours, planning periods, existence of breaks for employees, existence of part-time employees in addition to full-time ones, etc. TGAP has its natural tendency to be a multi-objective assignment problem where the objectives tended to have conflicting issues among them. For example, to have a better service quality, some service center needs to consider faster guiding time with an ideal number of visitors to be handled in a group/subgroup. To reach this ideal condition, it is often that the number of tour guides needs to be increased especially in a peak day/season. However, increasing the number of tour guides result in more cost for the service center. Since the service

center use a time-based system for paying salaries to additional tour guides according to their service hours, the most minimal cost for the service center will be preferred, thus adding another objective to be considered in addition to minimizing guiding time while keeping high visitor satisfaction.

In addition, since the process of assigning a tour guide to the tourists will highly influence the problem, it is necessary to optimize it from a multi-objective point of view to be as close as its natural behavior. However, different from single-objective optimization, it is impossible to find a single solution that can optimize all objectives because the problems involve multiple conflicting objectives. Improving one objective may at the same time degrade another one. Therefore, the solution will be at the best trade-off solutions which are important to decision-makers.

In this chapter, we would like to propose a multi-objective TGAP by considering other objectives in addition to the previous single-objective TGAP. Furthermore, we would like to compare the performance of several well-known multi-objective evolutionary algorithms to solve our proposed multi-objective tour guide assignment problem and evaluate the quality of solutions was evaluated by using representative performance indicators, which are hypervolume (HV), generational distance (GD), and maximum Pareto front error (MPFE).

3.2 Multi-objective TGAP Modeling

Generally, the TGAP consists of the assignment of a group of visitors to each available tour guide according to some restrictions and certain targeting goals. The problem was described as a single-service tourist center with several tour guides (*T*) who direct different visitor groups or sub-groups (*S*), where the guiding time is fixed for the guide to serve the visitor group, as shown in Figure 3.1. The main objective is to find the assignment that has the least total guiding time.

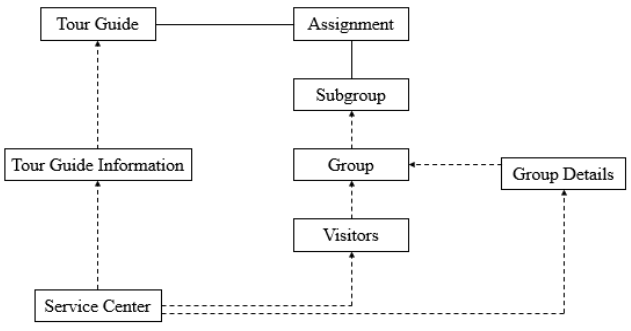


Figure 3.1 Unified Modelling Language of the TGAP Scheme

In this approach, the groups that must be guided constitute the set of tasks that must be accomplished. Tour guides have their own information as well as visitors and groups (Figure 3.2).

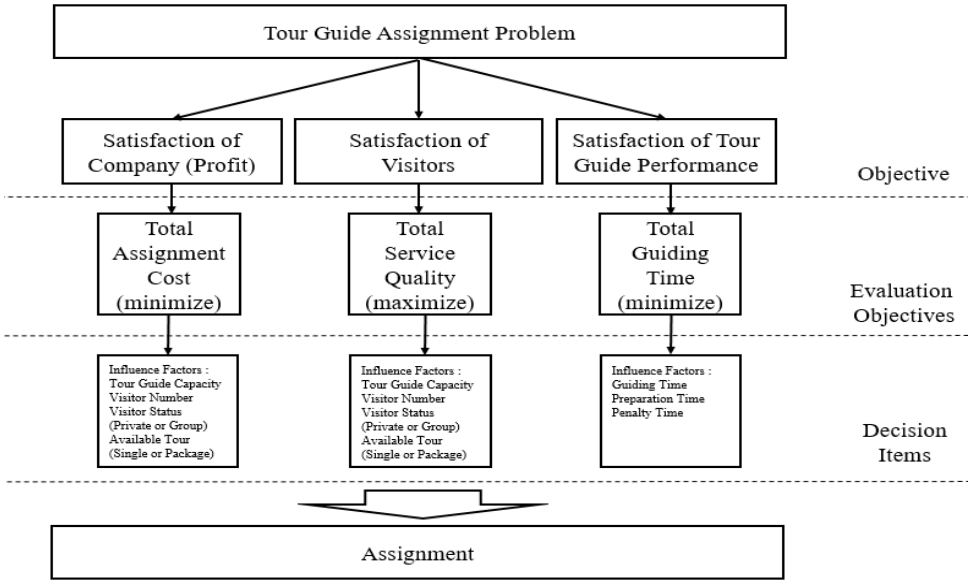


Figure 3.2 Proposed Multi-objective TGAP

Tour guides have a limited capacity to handle visitors, which is based on their skills. Highly skilled tour guides can guide more visitors than unskilled tour guides. To optimize the assignments, tour guides are assigned as many visitors as their capacity allows.

In this proposed model, three objectives are considered. In addition to the objective of minimizing total guiding time, we also considered two other objectives, which are minimizing total assignment costs and maximizing total service quality, as shown in Figure 3.2.

3.2.1 Minimize Total Guiding Time

As for the first objective, we set to minimizing total guiding time. Total guiding time is one of the most important factors in TGAP. Fast and effective total guiding time will increase the performance of the assignment of subgroups S into tour guide T.

Let $T = \{1, 2, \ldots, x\}$ for tour guides (Eq. 3.1)

Let $S = \{1, 2, \ldots, y\}$ for subgroups (Eq. 3.2)

For $i \epsilon T, j \epsilon S$ (Eq. 3.3)

Minimize $f_1(x) = \sum_{i=1}^{x} \sum_{j=1}^{y} \{G_{ij} V_{ij} + p_i q_i\}$ (Eq. 3.4)

$V_{ij} \epsilon \{0, 1\} \forall i \epsilon T, j \epsilon S$ (Eq. 3.5)

$q_i \epsilon \{0, 1\} \forall i \epsilon T$ (Eq. 3.6)

where G_{ij} represents the actual guiding time of the tour guide i, and V_{ij} is 1 when the tour guide i is assigned to a subgroup j and 0 otherwise. While p_i represents preparation time for the tour guide i, and q_i will have a value of 1 when the service sequence number of the former a subgroup j is higher and 0 otherwise.

3.2.2 Minimize Total Assignment Cost

As for the second objective, we set to minimizing total assignment cost. Total assignment cost is another important factor in TGAP. Every assignment of subgroup *S* into tour guide *T* has its own cost value since every tour guide has its own salary. Reducing the cost of assignment effectively will increase profit and balance.

Let $T = \{1, 2, \ldots, x\}$ for tour guides.

Let $S = \{1, 2, \ldots, y\}$ for subgroups.

For $i \epsilon T, j \epsilon S$

Minimize

$f_2(x) = \sum_{i=1}^{x} \sum_{j=1}^{y} \{P_{ij} V_{ij} + o_i t_i\}$ (Eq. 3.7)

$V_{ij} \epsilon \{0, 1\} \forall i \epsilon T, j \epsilon S$ (Eq. 3.8)

$o_i \epsilon \{0, 1\} \forall i \epsilon T$ (Eq. 3.9)

where P_{ij} represents the actual salary paid to the employees, and V_{ij} indicates a variable that takes a value of 1 for guide *i* when assigned to group *j*; otherwise, it is 0. While o_i represents overtime payment for the tour guide i, t_i will have a value of 1 when the tour guide has to work overtime; otherwise, it is 0.

3.2.3 Maximize Total Service Quality

Finally, as for the third objective, we set to maximizing total service quality. Total service quality is determined from the service quality that given by tour guide. It is also another important factor in TGAP. Depend on the number of visitors handled by the tour guide, the service might give different quality. Every assignment of subgroup *S* into tour guide *T* has its own service quality since every tour guide has its own salary. A high-quality service given by tour guides will increase the number of visitors.

Let $T = \{1, 2, \ldots, x\}$ for tour guides

Maximize

$$f_3(x) = \sum_{i=1}^{x} Q_i m_i \quad \text{(Eq. 3.10)}$$

$$m_i \epsilon \{0, 1\} \; \forall i \epsilon T \quad \text{(Eq. 3.11)}$$

where Q_i represents the quality of the service provided by tour guide i, and m_i has the value of 1 when the tour guide i is assigned to a subgroup j and 0 otherwise.

3.3 Well-known MOEA Approach to Solve MOTGAP

In the multi-objective problem, many considerations will be used and are involved in a decision-making process. Naturally, conflicts of interest among the objectives are unavoidable. As multi-objective evolutionary algorithms are a well-known approach to solving this class of complex problems, in this research, we used NSGAIII, NSGAII, ε-NSGA-I, ε-MOEA, the Pareto archived evolution strategy (PAES), the Pareto Envelope-based Selection Algorithm II (PESAII), and the Strength Pareto Evolutionary Algorithm II (SPEAII) to solve proposed MOTGAP model and evaluate the quality of obtained solutions by using representative performance indicators mentioned in previous chapter.

We first focused on defining the three objectives of the problem by considering real objectives that might have to be considered as in the real-life to represent this MOTGAP model. We looked for a set of Pareto optimal solutions that provides a series of suitable solutions. Each represented algorithm was performed and statistically analyzed by executing all represented algorithms in the MOEA framework to solve the assignment problem.

3.4 Computational Experiment

To generate a reference set of solutions for MOTGAP, different configuration settings for each algorithm have to be considered. Some of the parameters and configurations share common values and the same values for all algorithms, while some of them are specific to each algorithm as shown in Table 3.1.

Table 3.1 Parameters used for each algorithm in a comparative study

Parameter	Description	Algorithm
Population Size	size of the population	All Algorithms
sbx.Rate	simulated binary crossover rate	
sbx.DistributionIndex	simulated binary crossover distribution index	
1x.Rate	single-point crossover rate (for binary encoding)	
bf.Rate	bit-flip mutation rate (for binary encoding)	
epsilon	ε-dominance archive value	ε-NSGAII, ε-MOEA
injectionRate	population injection rate for restarting the evaluations	ε-NSGAII, ε-MOEA
bisections	number of bisections in the adaptive grid archive	PAES, PESAII
archiveSize	size of the archive	PAES, PESAII, SPEAII

In this study, we used different parameters value for each algorithm, which were set to be the best conditions of each algorithm based on the literature and default settings in the MOEA framework. Different configuration settings for each algorithm were considered and set as follows:

Table 3.2 Parameter settings used to generate P_{true} or the reference set

Parameter	Minimum Value	Maximum Value
Max Evaluation	2000	6000
Population Size	10	100
sbx.Rate	0.0	1.0
sbx.DistributionIndex	0.0	500.0
epsilon	0.0	1.0
injectionRate	0.1	1.0
1x.Rate	0.0	1.0
bf.DistributionIndex	0.0	500
archiveSize	10	1000

Also, to find out whether the MOEA default settings are the best parameter settings for the experiment before we started the experiment we vary the crossover value (sbx.Rate) and the mutation value (pm.Rate) at 0.0, 0.05, 0.1. The quality of solutions was evaluated by using representative performance indicators, which are hypervolume, generational distance, and maximum Pareto front error. These performance indicators are used to compare the algorithms and evaluate them statistically.

Finally, the comparative study of the algorithms was conducted based on the true Pareto optimal solution set generated by applying all algorithms to MOTGAP. The problem was then evaluated by considering all baseline algorithms with optimum configuration settings. According to Fujita et al., it is recommended to perform the evaluation using at least 50 seeds.

The data used in this experiment are set of artificial data, generated by using 25 tour guides with 500 visitors to be handled. Once again, the quality of solutions will be evaluated by using representative performance indicators mentioned above.

In addition, the algorithm with the best performance will further be evaluated for handling visitors with a large number of visitors in the group. Three cases were set with three different numbers of visitors: 500, 1000 and 1500.

3.4.1 Varying Crossover Rate (sbx.Rate)

As the beginning, the crossover value (sbx.Rate) was varied at 0.0, 0.5 and 1.0. While for other parameters, we used default settings in MOEA, 15 for the sbx distribution index, 0.5 for the mutation rate, 20 for the pm distribution index, and 100 for the population size.

Table 3.3 Sbx rate variation results for each algorithm in a comparative study

a. Hypervolume

Sbx.rate	0.0	0.5	1.0
NSGAII	0.29	0.31	0.35
ε-NSGAII	0.32	0.36	0.38
NSGAIII	0.38	0.43	0.45
ε-MOEA	0.37	0.41	0.41
PAES	0.13	0.15	0.18
PESAII	0.20	0.22	0.13
SPEAII	0.06	0.08	0.09

b. Generational Distance

Sbx.rate	0.0	0.5	1.0
NSGAII	0.030	0.023	0.018
ε-NSGAII	0.030	0.026	0.020
NSGAIII	0.045	0.042	0.036
ε-MOEA	0.028	0.033	0.025
PAES	0.070	0.060	0.030
PESAII	0.014	0.012	0.008
SPEAII	0.910	0.880	0.810

c. Maximum Pareto Front Error

Sbx.rate	0.0	0.5	1.0
NSGAII	0.37	0.30	0.26
ε-NSGAII	0.33	0.29	0.24
NSGAIII	0.29	0.24	0.20
ε-MOEA	0.15	0.12	0.13
PAES	0.32	0.26	0.23
PESAII	0.11	0.08	0.06
SPEAII	0.044	0.042	0.038

Judging from the results shown in Table 3.3, a suitable crossover rate is 1.0 for all algorithms. In detail, we can see that NSGA-III has better performance in terms of the spread of solutions along the Pareto front which showed the largest value on Hypervolume evaluation compared to others. However, in this experiment, on General Distance evaluation NSGAII has better performance while for Maximum Pareto Front Error evaluation, ε-MOEA was able to overcome other algorithms. Since all algorithm gives better performance at suitable mutation rate 1.0, then we set this value set for all next experiment.

3.4.2 Varying Mutation Rate (pm.Rate)

Based on the previous experiment, a suitable crossover rate was set at 1.0, and for the other parameters, we used default settings in MOEA, 15 for the sbx distribution index, 20 for the pm distribution index, and 100 for the population size.

Table 3.4 pm.Rate variation results for each algorithm in a comparative study

a. Hyper Volume

pm.Rate	0.0	0.05	0.1
NSGAII	0.29	0.35	0.34
ε-NSGAII	0.33	0.38	0.35
NSGAIII	0.39	0.45	0.41
ε-MOEA	0.32	0.41	0.37
PAES	0.10	0.15	0.14
PESAII	0.08	0.13	0.10
SPEAII	0.05	0.09	0.07

b. Generational Distance

pm.Rate	0.0	0.05	0.1
NSGAII	0.023	0.030	0.027
ε-NSGAII	0.026	0.030	0.029
NSGAIII	0.022	0.024	0.021
ε-MOEA	0.025	0.028	0.025
PAES	0.030	0.070	0.060
PESAII	0.004	0.014	0.010
SPEAII	0.840	0.910	0.900

c. Maximum Pareto Front Error

pm.Rate	0.0	0.05	0.1
NSGAII	0.37	0.28	0.31
ε-NSGAII	0.33	0.24	0.30
NSGAIII	0.17	0.13	0.15
ε-MOEA	0.27	0.20	0.22
PAES	0.33	0.23	0.29
PESAII	0.18	0.12	0.15
SPEAII	0.50	0.38	0.44

Judging from the results, a suitable mutation rate is 0.05 for all algorithms. Interestingly, the results showed, compared to other algorithms and regardless of the crossover rate value, NSGA-III is generally better in performance. A suitable mutation rate at 0.05 was then set for all next experiment.

Furthermore, the properties of the MOEAs observed while generating the approximation solution sets for MOTGAP were also analyzed. As a result, we could see that the number of solutions in the approximation solution set was generated by each MOEA used in this study as shown in Figure 3.3. In addition, their running times were also can be measured and are shown in Figure 3.4.

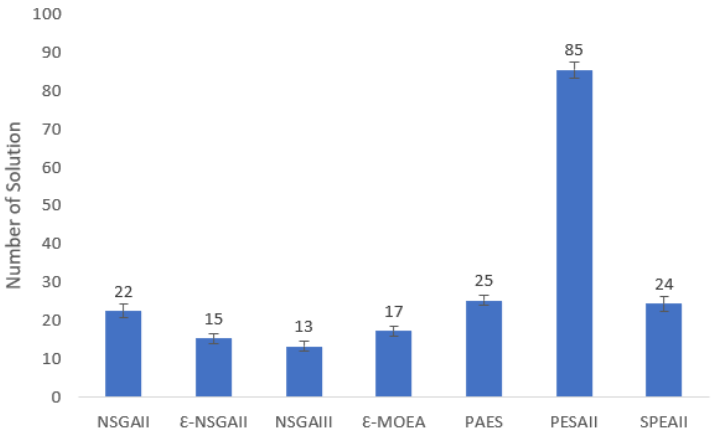


Figure 3.3 Number of Solutions Obtained for Each Algorithm on MOTGAP

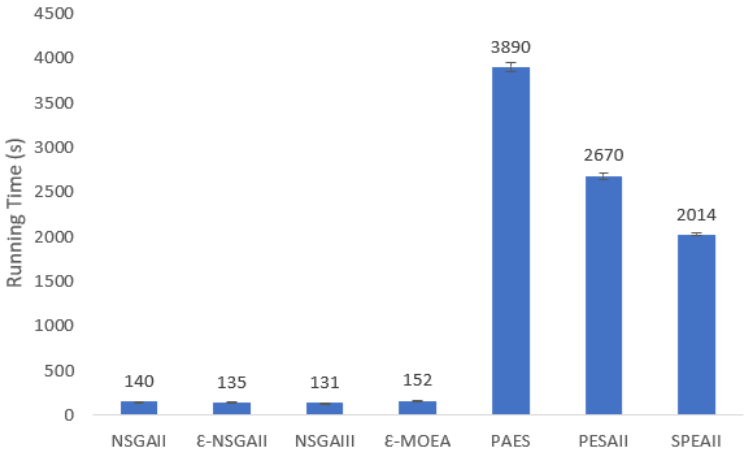


Figure 3.4 Running Time of Each Algorithm on MOTGAP

The results showed that PESA-II gives more solutions, compared to the others (Figure 3.3), however, the performance was not considered as good due to the long-running time (Figure 3.4). Also, three algorithms (ε-MOEA, ε-NSGA-II, and NSGA-III) have the best running times and performance, compared to the other algorithms for the TGAP. Based on this result, we selected these algorithms as a control problem and the main consideration for further analysis.

3.4.3 Set of Pareto Optimal Solutions

To get the set of obtained solutions, we combined all solutions into 20 runs and removed the dominant solutions. A set of Pareto optimal solutions comprising ε-MOEA, ε-NSGAII, and NSGAIII were then obtained, and they provide a series of feasible optimal solutions. The set of Pareto optimal solutions obtained from NSGAIII is shown in Figure 3.5. Compared to ε-MOEA and ε-NSGAII, NSGAIII was able to generate a large number of very different solutions to the TGAP.

In addition, NSGAIII also has the ability to maintain diversity among population members, supplying and adaptively updating several well-spread reference points, most of which are away from the preferred central vector. Meanwhile, the solutions obtained by ε-MOEA and ε-NSGAII showed a lack of diversity where the solutions were distributed very closely to the preferred central vector.

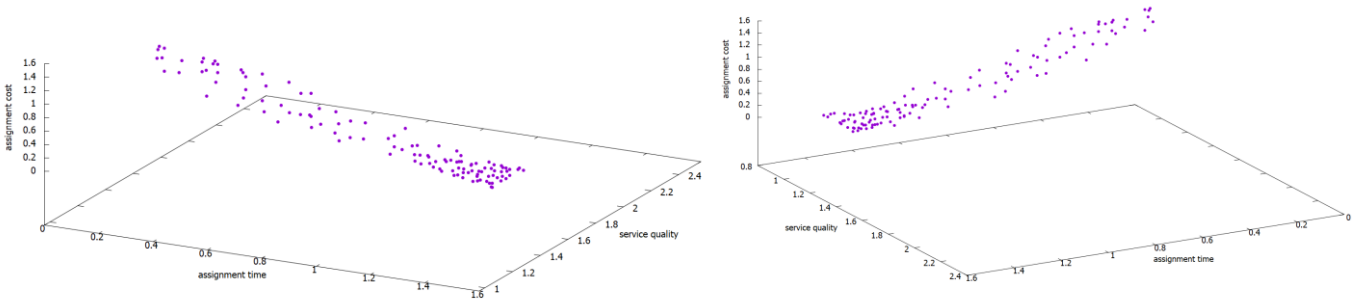


Figure 3.5 Set of Pareto Optimal Solutions of NSGAIII on MOTGAP

Furthermore, we compared the surface area of the obtained solutions between *ε*-MOEA–NSGAIII and *ε*-NSGAII–NSGAIII, as shown in Figures 3.6 and 3.7 respectively.

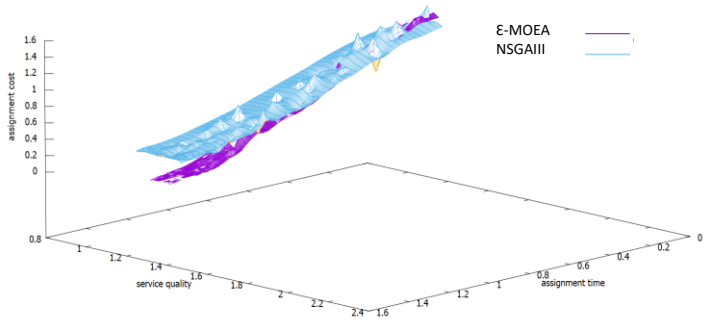


Figure 3.6 Surface Comparison of ε-MOEA and NSGAIII

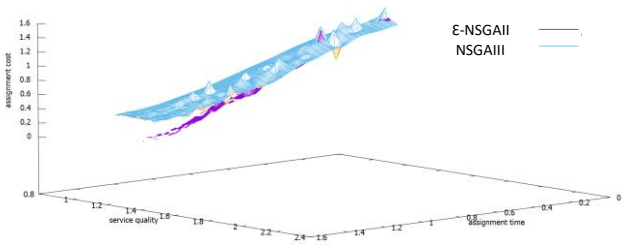
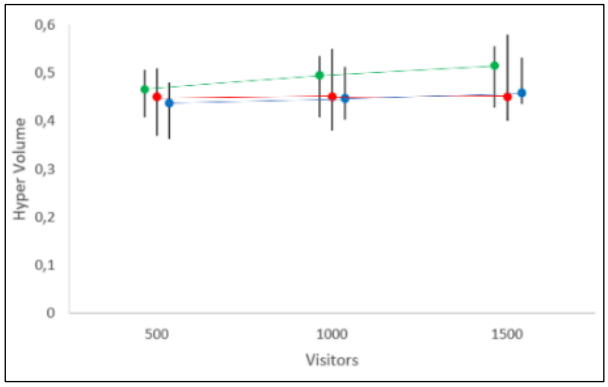


Figure 3.7 Surface Comparison of ε-NSGAII and NSGAIII

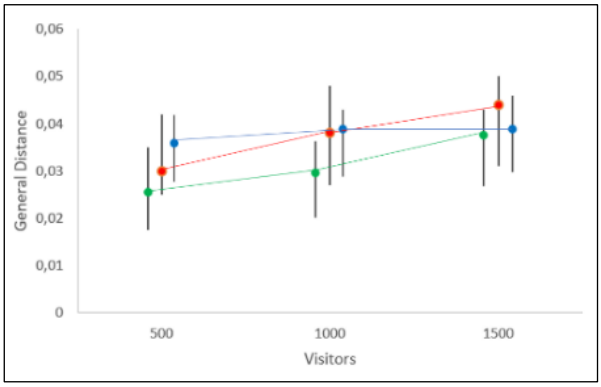
A Pareto front surface comparison showed that NSGAIII has a larger coverage area of reference points, compared to ε-MOEA and ε-NSGAII, indicating that the non-dominant solution's approximation set for NSGAIII covers the whole extension of the true Pareto front. This also indicates that NSGAIII tends to generate more widely spread candidate solutions.

3.4.4 The effect of Changing the Problem Scale

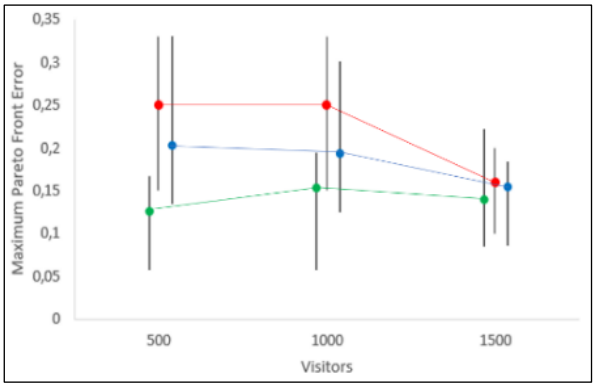
Furthermore, to evaluate the performance of NSGAIII, we compared it with other algorithms in three different cases: a MOTGAP with 500 visitors, with 1000 visitors, and with 1500 visitors. We used the same selected performance indicators as in the reference cases to compute and statistically analyze the results. As with the base case, we started with 500 visitors to obtain the performance indicator value of each algorithm, as seen in Figure 3.8.



(a)



(b)



(c)

Figure 3.8 Performance Indicator Comparison Results in Cases 1, 2, and 3 for Hyper Volume (a), Generational Distance (b) and Maximum Pareto Front Error (c) with ε-MOEA (blue), ε-NSGAII (red) and NSGAIII (green).

Since the results are not different in the smaller case (Figure 3.8a), we further increased the visitor numbers to have a larger problem (Figure 3.8b and 3.8c). Based on the results, we found that three algorithms (ε-MOEA, ε-NSGAII, and NSGAIII) were able to show their ability in solving MOTGAP any problem scale.

Furthermore, we can see that the median values of NSGAIII showed higher value on the hypervolume and smaller value generational distance indicators illustrating that NSGAIII has a greater diversity of solutions compared to the other algorithms. Also, NSGAIII has a smaller maximum Pareto front error value compared to the other algorithms, showing the ability of NSGAIII to generate an approximation set that is not far from the reference set.

However, the increase in the case numbers seems to not have much effect on the performance of NSGAIII, which might indicate that NSGAIII is also able to handle large-scale multi-objective problems.

3.5 Summary

In this research, we realized that it is natural for TGAP, as a real-world problem, to consider more than one objective. We were able to propose a multi-objective TGAP concept by considering two additional objectives, minimizing assignment cost and maximizing service quality, in addition to minimizing assignment time as in previous single objective TGAP.

Furthermore, we were able to apply some of well-known algorithm designed for solving the multi-objective problem and evaluate the performance of each algorithm on our proposed

multi-objective TGAP. Since in finding the optimum solution for proposed multi-objective TGAP, the algorithm must simultaneously optimize several conflicting objectives mentioned before, there will be no single point is considered as an optimal solution. The improvement in one objective will cause at least another objective not being able to be optimized due to conflicting objectives among them. The obtained optimal solution can only be a set of non-dominated and trade-off solutions.

Based on the results, in a suitable mutation and crossover rates, NSGAIII has better performance, compared to the other algorithms, in terms of solution quality and running time. Furthermore, increasing the case size does not have much effect on NSGAIII's performance, showing the stability of NSGAIII in handling a larger-scale multi-objective problem.

Chapter 4 ε-NSGAIII

4.1 Introduction

NSGAIII is one of the recent effective reference-point-based many-objective optimization algorithms. It is an extension of NSGAII designed for solving many-objective optimization problems. The fundamental components of NSGAIII are similar to NSGAII algorithm. However, it has significant changes in its selection mechanism, where the population in NSGAIII is guided by multiple predefined structured reference points to preserve the diversity of offspring solutions. Due to its power in guiding solutions to any predefined direction, NSGAIII has been gaining more acceptance in solving real-world many-objective optimization problems.

In Chapter 3, we were able to show that NSGAIII gives better performance than other algorithms such as NSGAII, ε-NSGAII, epsilon MOEA (ε-MOEA), Pareto archived evolution strategy (PAES), and the Pareto Envelope-based Selection Algorithm II (PESAII) in terms of solution quality and running time on MOTGAP in the MOEA framework. However, in NSGAIII, early generations of the population are associated with few of the supplied reference points. Subsequently, when selecting the parent population, there is a likelihood of selecting

parents associated with the same reference point. This might lead to the generation of offspring solutions that are close to their parents resulting in a reduction of diversity.

The quality of the Pareto set can be determined by the convergence and the diversity of solutions. The convergence refers to the ability to minimize the distance of solutions to the optimal front, while the diversity refers to maximizing the distribution of solutions over the optimal front. The ability to have a satisfactory to balance between them is a real challenge in multi-objective optimization. ε-dominance is a concept that known can be used to maintain the diversity of the Pareto set. Considering this, in this chapter, we introduce ε-dominance concept into NSGAIII algorithm to increase the performance of NSGAIII on MOTGAP and evaluate the quality of solutions by using representative performance indicators, which are hypervolume (HV), generational distance (GD), and Reference Point Convergence (RPC).

4.2 ε-Dominance Concept

The concept is based on ε-dominance archiving introduced by Laumanns and Deb [51] where the user can specify the precision with which they want to obtain the Pareto-optimal solutions to a multi-objective problem, in essence giving them the ability to assign relative importance to each objective. This is accomplished by applying a grid (sized by user-specified ε values) to the search space of the problem. In other words, the whole objective space will be divided into hyper-boxes, where each box have ε_j size in the j-th objective. Figure 5.1 describes the concept of ε dominance where the solution P ε-dominates the entire ABCDA area, assuming it is a minimization process. However, the epsilon concept allows P to dominate only PECFP area.

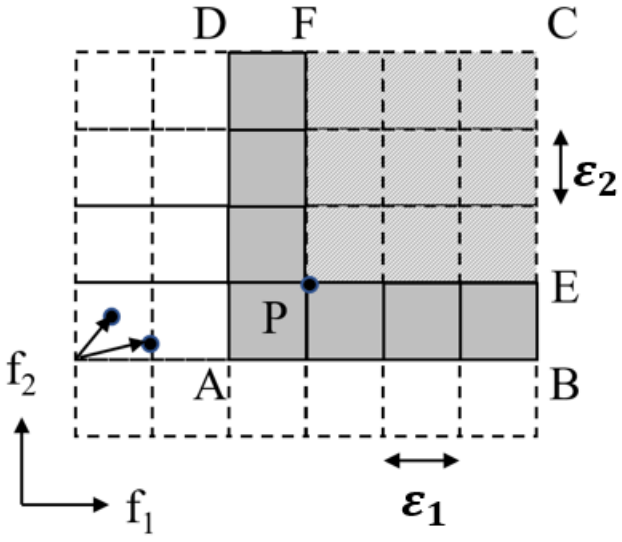


Figure 4.1 ε-dominance concept

Larger ε values result in a courser grid (and ultimately fewer solutions) while smaller ε values produce a finer grid. The fitness of each solution is then mapped to a fitness box based on the specified ε values.

Non-domination sorting is then conducted using each solution's box fitness, and solutions with identical box fitness (i.e., solutions that occur in the same grid block) are compared and those that are dominated within the grid block are eliminated.

This results in no more than one non-dominated solution existing in any one grid block, preventing clustering of solutions and promoting a more even search of the objective space. With this concept, it is possible to maintain a diverse Pareto set approximation and to prevent nondominated solutions from being lost. Compared to the original NSGAIII, ε-NSGAIII concept will generally look like this scheme.

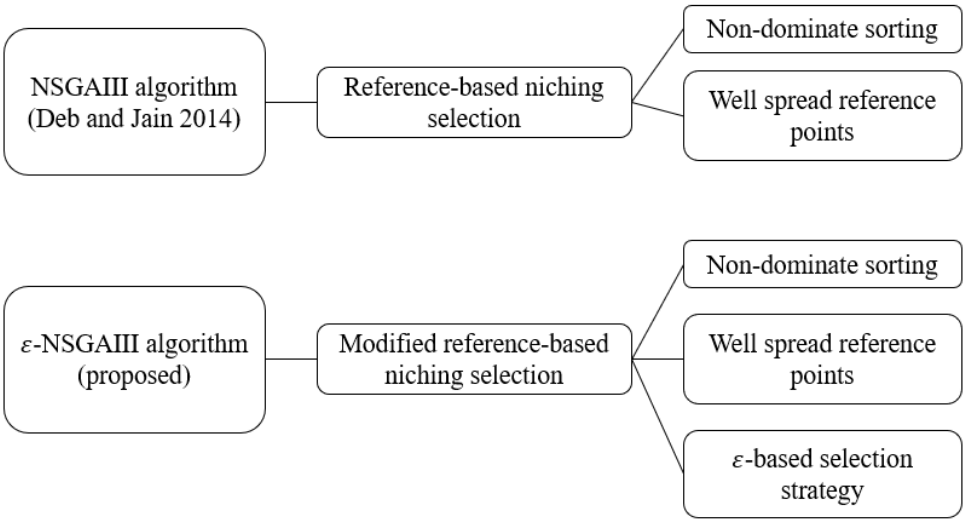


Figure 4.2 ε-NSGAIII Scheme

4.3 Proposed ε-NSGAIII Algorithm

Basically, the main components of ε-NSGAIII are similar to its original NSGAIII. The difference is in ε-NSGAIII the population was constructed from the random population combined with ε-non-dominated solutions kept at a constructed archive file. As the search progresses, the population size is automatically adapted based on the number of ε-non-dominated solutions that the algorithm has found. ε-non-dominated solutions found after each generation are stored in an archive and subsequently used to direct the search using a 25% injection scheme. ε-NSGAIII use the same dominance-based selection as NSGAII, which

called fast non-dominated sorting. After that, the best members from the last non-dominated front F_l in NSGAIII are selected based on the supplied reference points (Algorithm 4.1).

Algorithm 4.1 Pseudo code of ε-NSGAIII.
1. **procedure** ε-NSGAIII (N,N_A) 2. Define a set of reference points 3. Initialize population P_t (N) 4. Generate random population (size N_{pop}) 5. Store ε non-dominated solutions from P_t into an archive population A_t 6. **for** i = 1 *to generation number (t)* **do** 7. Create population R_t compose of 25% ε-non-dominated archive solutions and 75% generated random population. 8. Fast non dominated sorting $\mathcal{F}$ on population R_t 9. Sort population P_t and divide it into a number of non-dominated fronts (F_1, F_2, ..., F_l) 10. Continue the process until F_l is reached 11. Normalize the population 12. Associate with reference points 13. Apply the niche preservation mechanism 14. Apply ε non-dominated sorting 15. Keep the obtained solutions in the archive for the next generation 16. end **for**

4.4 Computational Experiments

Evolutionary algorithms perform well in finding multi-objective solutions and can ultimately obtain a non-dominated solution set. Therefore, how to evaluate the performance of these algorithms is also extremely important. There are two ways to judge the quality of non-dominated solution sets. First, after the decision space is projected into the target space, it is dependent on the distance between the retrieved PF and the actual PF surface. The smaller the distance, the better the convergence. Second, it depends on the distribution of the PF surface. The more uniform the distribution, the better the PF. The quality of solutions was then evaluated by using representative performance indicators, which are Hypervolume, Generational Distance, and Reference Point Convergence [60].

4.4.1 Parameters Setting

In order to find the most suitable Epsilon Coefficient (*C*) value for the experiment, we change *C* parameter in ZDT4 and ZDT1 test problems by using some random *C* parameter value to test and observe a different range of setting. We first use *C* value at 1, 25 and 50 which represent small, average and large value respectively. Then based on the result, we evaluate the algorithm by using approximated *C* at a different value. Finally, the suitable parameters were applied to MOTGAP. All algorithms in this experiment were run following all key parameters as shown in the table below. To evaluate the quality of solutions, Hypervolume, General Distance and

Reference Point Convergence were being used as performance indicators to compute and statistically analyzed the results.

Table 4.1 Parameters used for each algorithm in comparative study

	Preliminary evaluation	Final evaluation
maximum number of simulations	5000/ 10000/ 20000	10000 (ZDT4) and 5000 (ZDT1)
number of replications	1	10
population size	50	50
number of candidates	50	50
child population size	50	50
number of initial reference points	1	1
reference point	(0.5, 0)	(0.5, 0)
select first fronts completely	false	false
guidance reference points distance metric	ASF	ASF
cluster representative choice	best	best
epsilon coefficient	0.001/1/25/50	0.001/ 0.1/ 0.2/ 0.3
crossover operator for continuous variables	SBX	SBX
crossover operator for discrete variables	uniform crossover	uniform crossover
crossover probability	0.8	0.8
mutation probability	0.07	0.07
reproduction selection operator	tournament	tournament

4.4.2 Evaluation on Benchmark function: ZDT4

As a preliminary experiment, we applied the proposed ε-NSGAIII and original NSGAIII on ZDT4 problem at different C values as seen in the parameter table and measure the performance with Hypervolume, Generational Distance and Reference Point Convergence. The results were then being compared. We run a preliminary experiment on 5000 evaluation runs, with a variation of C value at 1, 25 and 50 as shown in Table 4.2.

The quality of the solutions was first evaluated with a Hypervolume performance indicator. The result showed that increasing the C value influences the result. Interestingly, despite the change of C value, ε-NSGAIII gave better result compare to NSGAIII at any C value. As we can see here in Table 4.2, higher hypervolume value showed the best result was obtained at C=1.

Table 4.2 Hypervolume value on 5000 evaluation runs

Algorithm	Hypervolume		
	Maximum	Average	Minimum
NSGAIII	0.9748	0.9293	0.5832
ε-NSGAIII, $C1$=1	0.9891	0.9521	0.6218
ε-NSGAIII, $C2$=25	0.9773	0.9356	0.6398
ε-NSGAIII, $C3$=50	0.9897	0.9348	0.6302

In addition, we also run the experiment at 20000 (Table 4.3) evaluation runs. From the result, we knew that epsilon NSGAIII still have a better result than the original NSGAIII. However, as the number of evaluations increased, we found that the mean value of the results was decreased at any *C* value, as shown in Table 4.3. This might be because the result becomes downgrade during long-running, which causing the quality of the obtained solution to be decreased.

Table 4.3 Hypervolume value on 20000 evaluation runs

Algorithm	Hypervolume		
	Maximum	Average	Minimum
NSGAIII	0.9827	0.9305	0.5789
ε-NSGAIII, *C1*=1	0.9931	0.9519	0.5512
ε-NSGAIII, *C2*=25	0.9835	0.8949	0.5265
ε-NSGAIII, *C3*=50	0.9890	0.9022	0.5731

Based on these results, we decide to use 10000 evaluation runs for final analysis, expecting that the result might not too early to finish or too long that might give result degradation. The best *C* value at both preliminary experiments was obtained at *C*=1, indicating that the most suitable *C* value might be less than 1. We then decide to vary *C* value at 0.01, 0.1, 0.2 and 0.3. As a result, the highest value was obtained at *C2*=0.1 (Table 4.4) showing the most suitable *C* value for the algorithms on ZDT4 problem.

Table 4.4 Hypervolume value on 10000 evaluation runs

Algorithm	Hypervolume		
	Maximum	Average	Minimum
NSGAIII	0.9723	0.9221	0.5540
ε-NSGAIII, *C1*=0.01	0.9883	0.9058	0.0190
ε-NSGAIII, *C2*=0.1	0.9867	0.9102	0.0483
ε-NSGAIII, *C3*=0.2	0.9813	0.9041	0.0554
ε-NSGAIII, *C4*=0.3	0.9827	0.9002	0.0601

Similarly, we also evaluate the performance of proposed ε-NSGAIII with the General Distance performance indicator. Based on the previous preliminary experiment, we set the parameter as 10000 evaluation runs, with a variation of *C* value at 0.01, 0.1, 0.2 and 0.3 as shown in Table 4.5. From the results, once again we can see that the best result obtained at *C2*=0.1, which showed by smaller Generational Distance values compared to other sets. In this experiment all ε-NSGAIII also gave better results compared to its original NSGAIII.

Table 4.5 Generational Distance value on 10000 evaluation runs

Algorithm	Generational Distance		
	Maximum	Average	Minimum
NSGAIII	0.5853	0.1928	0.0965
ε-NSGAIII, *C1*=0.01	0.5727	0.1839	0.0912
ε-NSGAIII, *C2*=0.1	0.5727	0.1817	0.0885
ε-NSGAIII, *C3*=0.2	0.5727	0.1891	0.0903
ε-NSGAIII, *C4*=0.3	0.5727	0.1860	0.0843

In order to see the convergence of the obtained solution to the reference point, we evaluate the solution by using Reference Point Convergence performance indicator. We set the parameters as 10000 evaluation runs, with variation of *C* value at 0.01, 0.1, 0.2 and 0.3. From the results, we found that increasing the *C* value could increase the average obtained value, as shown in Table 4.6 below. However, differently, the most suitable *C* value for ε-NSGAIII was obtained at C*1*=0.01.

Table 4.6 Reference Point Convergence value on 10000 evaluation runs

Algorithm	Reference Point Convergence		
	Maximum	Average	Minimum
NSGAIII	1.7188	0.8996	0.5714
ε-NSGAIII, *C1*=0.01	1.7423	0.9982	0.5120
ε-NSGAIII, *C2*=0.1	1.7423	1.0573	0.5856
ε-NSGAIII, *C3*=0.2	1.7423	1.1512	0.5275
ε-NSGAIII, *C4*=0.3	1.7423	1.2690	0.6107

4.4.3 Evaluation on Benchmark function: ZDT1

Furthermore, we used ZDT1 problem and applied proposed ε-NSGAIII to see the effect of changing *C* value and measure the performance quality with Hypervolume, Generational Distance and Reference Point Convergence. The result was then compared to the original NSGAIII. For the preliminary experiment, the evaluation run was set at 5000 and the *C* value was set at 1, 25 and 50 as shown in Table 4.7.

At first, the performance was evaluated using Hypervolume performance indicator. As a result, we found that increasing the *C* value influences the result. One more time, all ε-NSGAIII gave better result compare to NSGAIII. However, the best suitable *C* value for ε-NSGAIII was obtained at *C2*=25. Indicating that the best parameter may be laid between *C1*=1 and *C2*=25, as at *C3*=50 the result was decreased.

Table 4.7 Hypervolume value on 5000 evaluation runs

Algorithm	Hypervolume		
	Maximum	Average	Minimum
NSGAIII	0.9697	0.9264	0.6104
ε-NSGAIII, *C1*=1	0.9735	0.9478	0.6354
ε-NSGAIII, *C2*=25	0.9773	0.9580	0.6322
ε-NSGAIII, *C3*=50	0.9802	0.9441	0.6347

To avoid early or too long evaluation run which might decrease the result, we decide to use 10000 evaluation runs directly for final analysis. Also, the *C* value was set at 0.01, 5, 10. As the results, similar in ZDT4 problem, ε-NSGAIII showed better results than the original NSGAIII in ZDT1 as shown in Table 4.8. The best result for ZDT1 problem was obtained at *C4*=15 when evaluating with Hypervolume performance indicator.

Table 4.8 Hypervolume value on 10000 evaluation runs

Algorithm	Hypervolume		
	Maximum	Average	Minimum
NSGAIII	0.9673	0.9258	0.5904
ε-NSGAIII, *C1*=0.01	0.9790	0.9365	0.0174
ε-NSGAIII, *C2*=5	0.9831	0.9389	0.3879
ε-NSGAIII, *C3*=10	0.9928	0.9457	0.4271
ε-NSGAIII, *C4*=15	0.9844	0.9486	0.5843

Similarly, we also evaluate the performance of proposed ε-NSGAIII in with General Distance performance indicator. Based on the previous preliminary experiment, we set the parameter at 10000 evaluation runs, with a variation of *C* value at 0.01, 5, 10 and 15 as shown in Table 4.9.

From the results, once again we can see that the best result obtained at *C2*=5, which showed by smaller Generational Distance values compared to other sets. In this experiment all ε-NSGAIII also gave better results compared to its original NSGAIII.

Table 4.9 Generational Distance value on 10000 evaluation runs

Algorithm	Generational Distance		
	Maximum	Average	Minimum
NSGAIII	0.5729	0.1858	0.0953
ε-NSGAIII, *C1*=0.01	0.5713	0.1880	0.0892
ε-NSGAIII, *C2*=5	0.5713	0.1862	0.0857
ε-NSGAIII, *C3*=10	0.5713	0.1873	0.0901
ε-NSGAIII, *C4*=15	0.5713	0.1899	0.0915

Finally, in this experiment, similar to ZDT4 problem, we found that increasing the *C* value on proposed ε-NSGAIII could increase the average obtained value in ZDT1, as shown in Table 4.10. We set the parameters as 10000 evaluation runs, with a variation of *C* value at 0.01, 1, 5 and 15.

Table 4.10 Reference Point Convergence value on 10000 evaluation runs

Algorithm	Reference Point Convergence		
	Maximum	Average	Minimum
NSGAIII	1.6502	0.8876	0.5151
ε-NSGAIII, *C1*=0.01	1.7259	0.9516	0.5092
ε-NSGAIII, *C2*=5	1.7259	0.9827	0.5511
ε-NSGAIII, *C3*=10	1.7259	1.095	0.5478
ε-NSGAIII, *C4*=15	1.7259	1.174	0.5832

From both results, ZDT4 and ZDT 1, we found that different problem has its own optimum *C* value. However, changing the *C* value did not change the domination of ε-NSGAIII over the original NSGAIII. So, as for the next experiment we tried to apply and compare proposed ε-NSGAIII to MOTGAP.

4.4.4 Performance Measurement of Proposed ε-NSGAIII

Based on varying *C* value on test problem experiments, we set C value into *C*=1 with 10000 evaluation and evaluate the result with Hypervolume, Generational Distance and Reference Point Convergence. Also, as in the previous chapter, the data used in this experiment are set of artificial data, generated by using 25 tour guides with 500 visitors to be handled. Series of box plot (or box and whisker diagram) are being used to display the distribution of data based on the five numbers summary: minimum, first quartile, median, third quartile, and maximum. In the simplest box plot, the central rectangle spans the first quartile to the third quartile (the interquartile range or IQR). A segment inside the rectangle shows the median, and whiskers above and below the box show the locations of the minimum and maximum.

As mention, the Hypervolume indicator showed the measure of the spread of solutions along the Pareto front, as well as the distance of the set from the Pareto-optimal front. According to the experiment set by Deb [41], among the algorithm, the one with the largest value of the Hypervolume has better performance. We can see from the result shown in Table 4.11 and

Figure 4.3, ε-NSGAIII has a higher value and outperforms other algorithms. In other words, proposed ε-NSGAIII has a better performance compared to other algorithms.

Table 4.11 Comparative evaluation of Hypervolume result on MOTGAP

Algorithm	Hypervolume		
	Maximum	Average	Minimum
NSGAII	0.60	0.31	0.22
ε-NSGAII	0.51	0.36	0.24
NSGAIII	0.61	0.43	0.27
ε-NSGAIII	0.59	0.49	0.25
ε-MOEA	0.61	0.41	0.25
PAES	0.28	0.15	0.09
PESAII	0.30	0.22	0.15
SPEAII	0.15	0.08	0.02

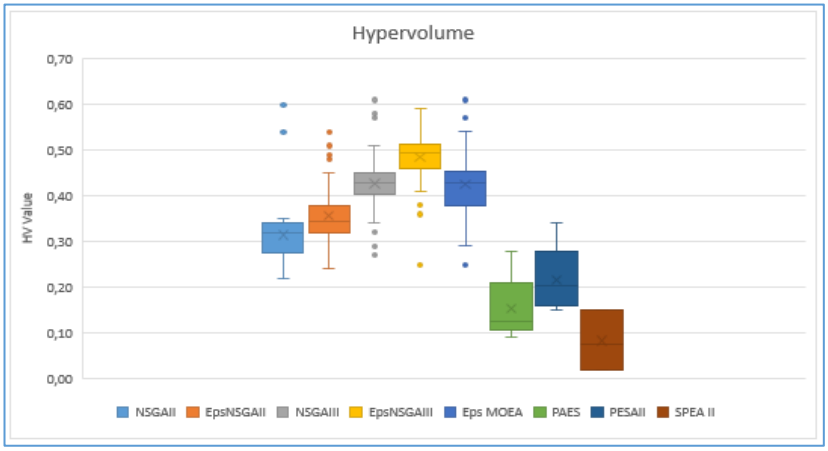


Figure 4.3 Comparative evaluation of Hypervolume result on MOTGAP

The generational distance is proposed by Veldhuizen & Lamont to estimate the distance between any individual of the obtained Pareto front and the global Pareto front. The lower the value indicates that the algorithm has better performance since the obtained Pareto front has a closer distance with the global Pareto front. From the value shown in Table 4.12 and Figure 4.4, ε-NSGAIII showed lower generational distance compared to other algorithms, indicating a better convergence on obtained results.

Table 4.12 Comparative evaluation of Generational Distance result on MOTGAP

Algorithm	Generational Distance		
	Maximum	Average	Minimum
NSGAII	0.08	0.033	0.022
ε-NSGAII	0.07	0.030	0.024
NSGAIII	0.05	0.025	0.020
ε-NSGAIII	0.05	0.019	0.009
ε-MOEA	0.08	0.031	0.021
PAES	0.11	0.070	0.020
PESAII	0.26	0.130	0.040
SPEAII	0.91	0.780	0.340

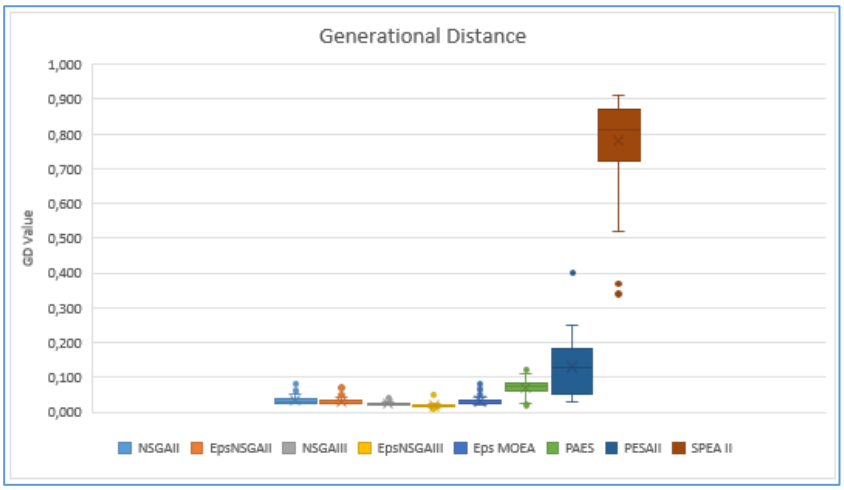


Figure 4.4 Comparative evaluation of Generational Distance result on MOTGAP

From these experiences, we found that ε-NSGAIII overcome other algorithms on TGAP, including the original NSAIII, showing that applying epsilon concept on NSGAIII increases the performance of proposed algorithm.

Reference Point Convergence indicator can be applied specifically for the performance evaluation of the reference point-based algorithms. According to Siegmund, the smaller is the value of the metric, the better is the speed of convergence. Once again, from Table 4.13 and Figure 4.5, we can see that ε-NSGAIII outperform other algorithms in this performance indicator.

Table 4.13 Comparative evaluation of Reference Point Convergence result on MOTGAP

Algorithm	Reference Point Convergence		
	Maximum	Average	Minimum
NSGAII	0.063	0.048	0.022
ε-NSGAII	0.063	0.045	0.026
NSGAIII	0.056	0.038	0.022
ε-NSGAIII	0.051	0.035	0.027
ε-MOEA	0.085	0.061	0.025
PAES	0.125	0.065	0.037
PESAII	0.160	0.090	0.030
SPEAII	0.190	0.071	0.030

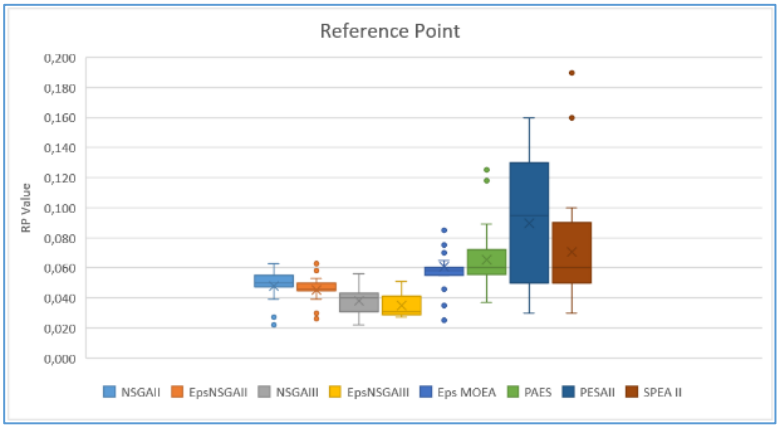


Figure 4.5 Comparative evaluation of Reference Point Convergence result on MOTGAP

From these experiences, we found that ε-NSGAIII overcome other algorithm on TGAP, including the original NSAIII, showing that applying epsilon concept on NSGAIII increase the performance of proposed algorithm.

4.5 Summary

In this research, we were able to evaluate ε-NSGAIII performance on a multi-objective Tour Guide Assignment Problem, compared with other well-known evolutionary algorithms. Based on the results, increasing the value of C of proposed algorithm will increase the performance of the algorithm, however, the proposed algorithm still be better than others nevertheless the changing in C value. Also, ε-NSGA-III has better performance, compared to other algorithms, at C value=1.

Chapter 5 Adaptive ε-NSGAIII

5.1 Introduction

As described in Chapter 4, we were able to show the ability of proposed ε-NSGAIII in giving a better solution compared to other well-known MOEAs when solving MOTGAP. The main additional feature in ε-NSGAIII compared to its original NSGAIII is the use of ε parameter, which can be set before solving the problem to obtain a good solution. According to Sigmund, et al. [57], when we select higher or lower values of ε, the behavior of the algorithm will be changed, giving a wider or narrower region of the feasible space so we can select the desired non dominated solutions.

As a consequence, this will change the results in a set of more or less diverse solutions, which are close to the reference point. Also, applying higher values of ε will give us a better understanding of the solutions, but this will decrease searching speed. On the other hand, applying lower values of ε will make the algorithm works faster. However, a better understanding of the solutions cannot be obtained, especially in the earliest generations of the algorithm.

For this reason, to obtain a better solution with a better understanding of the solutions we need an ε value that has the ability to change the static concept of ε parameter. In this chapter, we introduce extended ε-NSGAIII algorithm by introducing adaptive epsilon concept

to improve the performance of ε-NSGAIII on multi-objective tour guide assignment problem and a well-known real-life problem named Multi-objective Travelling Salesman Problem (MOTSP). Furthermore, we evaluate the quality of solutions by using representative performance indicators, which are Hypervolume (HV), Generational Distance (GD), Reference Point Convergence (RPC) and Spread Evaluation.

5.2 Adaptive Approach

The adaptive approach was made based on the work by Siegmund, et al. [62] with modification on the reference point behavior. Instead of using the static ε parameter, it will use an adaptive approach as shown in the picture. In the adaptive version, it will automatically calculate and adjust the ε parameter value proportional to the distance between the closest solution and distributed reference point. In summary, the difference between the static approach and adaptive one will be as below.

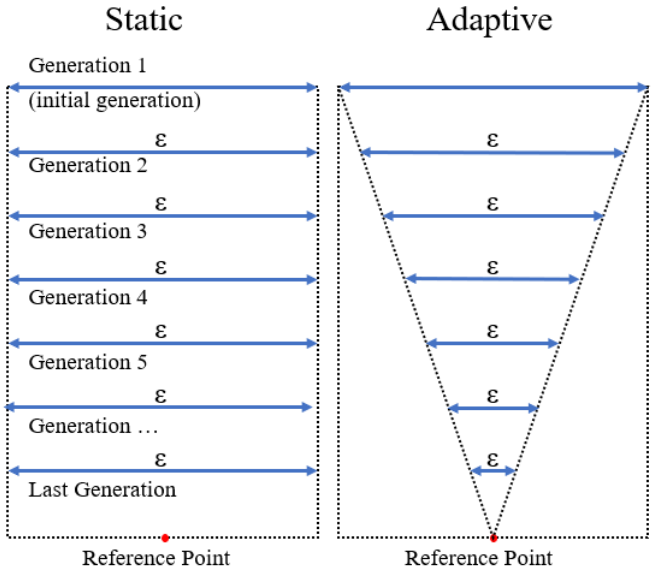


Figure 5.1 Static and adaptive approach for ε-parameter

5.3 Adaptive ε-NSGAIII Algorithm

Basically, the mechanism in this method would be similar as ε-NSGAIII, the difference is in this proposed method, after a new set of population and offspring generates, it will be expected to be close to the desired reference point, then it will be sorted based on NSGAIII concept. ε-parameter value then automatically calculated and adjusted proportionally to the distance

between the closest solution and reference point. After that, with similar concept, when there is more than one reference point, based on the distance of solutions to reference points the algorithm clusters the set to some subsets; each subset consists of solutions which are closer to one specific reference point; then, the algorithm calculates different ε parameters based on the distance of the closest solution of each subset to the corresponding reference point (Algorithm 5.1).

Algorithm 5.1 Pseudo code of Adaptive ε-NSGA-III.
1. **procedure** A-ε-NSGAIII (N,N_A) 2. Define a set of reference points 3. Initialize population P_t (N) 4. Generate random population (size N_{pop}) 5. Generate ε value 6. Store ε non-dominated solutions from P_t into an archive population A_t 7. **for** i = 1 *to generation number (t)* **do** 8. Create population R_t compose of 25% ε-non-dominated archive solutions and 75% generated random population. 9. Fast non dominated sorting $\mathcal{F}$ on population R_t 10. Sort population P_t and divide it into a number of non-dominated fronts (F_1, F_2, ... F_l) 11. Continue the process until F_l is reached 12. Normalize the population 13. Associate with reference points 14. Calculate ε value and adjusted proportional to the distance between the closest solution and reference point 15. Keep the obtained ε value for the next generation 16. Apply the niche preservation mechanism 17. Keep the obtained solutions for the next generation 18. end **for**

In summary, the difference between ε-NSGAIII and the extended Adaptive ε-NSGAIII will be like this scheme below.

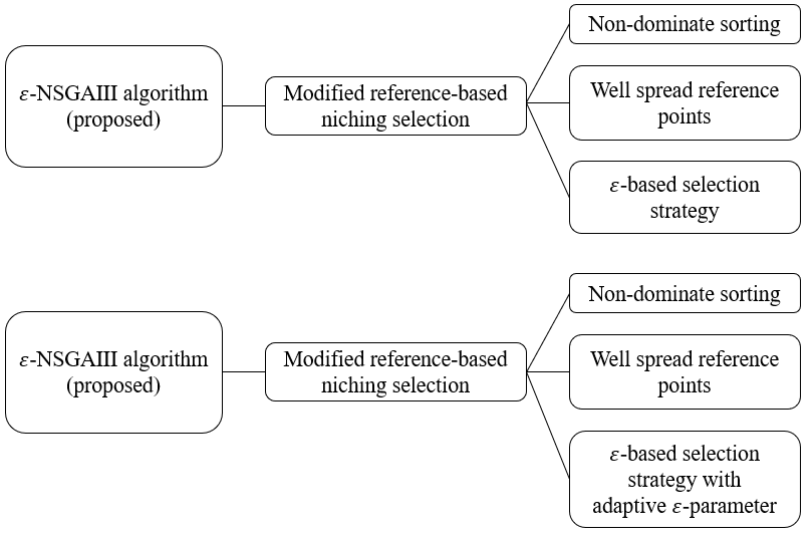


Figure 5.2 Adaptive ε-NSGAIII Scheme

The same as TGAP, MOTSP can be solved using conventional technique and Evolutionary based technique. However, from some published researches, it is known that Evolutionary based technique with its multi-objective evolutionary algorithms (MOEAs) are well-suited for solving several complex multi-objective problems with more objectives like MOTSP.

As for the experiment parameter, we use the same parameters as the previous experiment by using MOEA framework. The population size was set at 100 with 10000 generations. The results were compared based on Hypervolume, Generational-Distance, Reference-point Convergence and Spread Evaluation. The software was implemented in Java language and the use of MOEA framework for the conditions of experiments.

5.4 Proposed Adaptive ε-NSGAIII Performance on MOTGAP

Performance of proposed Adaptive ε-NSGAIII was measured to find out and compare the quality of solutions obtained from applying the proposed algorithm. Similar to previous ε-NSGAIII, the proposed adaptive ε-NSGAIII also will be evaluated by using representative performance indicators, which are Hyper Volume, Generational Distance, and Reference Point Convergence. Based on the previous experiment, in this experiment, we also first set C value into C=1 with 10000 evaluation, set of artificial data, generated by using 25 tour guides with 500 visitors to be handled and evaluate the result.

As mention before, the hypervolume indicator showed the measure of the spread of solutions along the Pareto front, as well as the distance of the set from the Pareto-optimal front.

Table 5.1 Comparative evaluation of Hypervolume result on MOTGAP

Algorithm	Hypervolume		
	Maximum	Average	Minimum
NSGAII	0.60	0.31	0.22
ε-NSGAII	0.51	0.36	0.24
NSGAIII	0.61	0.43	0.27
ε-NSGAIII	0.59	0.49	0.25
A ε-NSGAIII	0.61	0.50	0.22

As a result, Adaptive ε-NSGAIII showed a slightly higher value compare to ε-NSGAIII, while it outperforms other algorithms, as shown in Table 5.1.

The generational distance is proposed by Veldhuizen & Lamont, (1998), to estimate the distance between any individual of the obtained Pareto front and the global Pareto front. The lower the value indicates that the algorithm has better performance since the obtained Pareto front has a closer distance to the global Pareto front.

Table 5.2 Comparative evaluation of Generational Distance result on MOTGAP

Algorithm	Generational Distance		
	Maximum	Average	Minimum
NSGAII	0.080	0.033	0.022
ε-NSGAII	0.070	0.030	0.024
NSGAIII	0.050	0.025	0.020
ε-NSGAIII	0.050	0.019	0.009
A ε-NSGAIII	0.049	0.020	0.070

The generational distance indicator estimates the distance between any individual of the obtained Pareto front and the global Pareto front. However, Adaptive ε-NSGAIII showed a slightly lower value compare to ε-NSGAIII, while it outperforms other algorithms.

The reference point indicator showed that Adaptive ε-NSGAIII has a slightly\higher value on the maximum point compares to ε-NSGAIII, meanwhile, it outperforms other algorithms.

Table 5. 3 Comparative evaluation of Reference Point result on MOTGAP

Algorithm	Reference Point Convergence		
	Maximum	Average	Minimum
NSGAII	0.063	0.048	0.022
ε-NSGAII	0.063	0.045	0.026
NSGAIII	0.056	0.038	0.022
ε-NSGAIII	0.051	0.035	0.027
A ε-NSGAIII	0.048	0.035	0.026

Since the convergence of the obtained solution cannot differentiate the performance of the algorithm in solving TGAP, we tried to evaluate the obtained solutions in terms of diversity. Spread evaluation can quantify the performance of an algorithm from this point of view.

Table 5.4 Comparative evaluation of Spread Evaluation result on MOTGAP

Algorithm	Spread Evaluation		
	Maximum	Average	Minimum
NSGAII	0.982	0.903	0.893
ε-NSGAII	0.860	0.852	0.821
NSGAIII	0.899	0.875	0.836
ε-NSGAIII	0.872	0.843	0.804
A ε-NSGAIII	0.758	0.731	0.722

From the result we can see that obtained solutions from Adaptive ε-NSGAIII has the smallest value among other algorithms, including ε-NSGAIII and NSGAIII, showing that Adaptive ε-NSGAIII has better performance in terms of the diversity of obtained solutions compared to other tested algorithms.

5.5 Proposed Adaptive ε-NSGAIII Performance on MOTSP

As mentioned before, the hypervolume indicator showed the measure of the spread of solutions along the Pareto front, as well as the distance of the set from the Pareto-optimal front. The result showed that all algorithms can solve MOTSP clearly, where both ε-NSGAIII and Adaptive ε-NSGAIII has better performance compare to other algorithms (Table 5.5). However, the same as before, the result cannot show which one is better among the two proposed algorithms.

Table 5.5 Comparative evaluation of Hypervolume result on MOTSP

Algorithm	Hypervolume		
	Maximum	Average	Minimum
NSGAII	0.420	0.390	0.362
ε-NSGAII	0.432	0.411	0.396
NSGAIII	0.408	0.386	0.351
ε-NSGAIII	0.482	0.447	0.435
A ε-NSGAIII	0.494	0.452	0.438

In the generational distance, the lower the value indicates that the algorithm has better performance since the obtained Pareto front has a closer distance to the global Pareto front. From the result on the table, we can also see that obtained solution from both ε-NSGAIII and Adaptive ε-NSGAIII has the closest distance to the global Pareto Front, showing their better performance quality compare to other algorithms.

Table 5.6 Comparative evaluation of Generational Distance result on MOTSP

Algorithm	Generational Distance		
	Maximum	Average	Minimum
NSGAII	0.062	0.050	0.038
ε-NSGAII	0.058	0.042	0.040
NSGAIII	0.072	0.058	0.055
ε-NSGAIII	0.044	0.038	0.030
A ε-NSGAIII	0.042	0.035	0.025

The generational distance indicator estimates the distance between any individual of the obtained Pareto front and the global Pareto front. However, once again, Adaptive ε-NSGAIII showed a slightly lower value compare to ε-NSGAIII.

Reference point convergence indicator showed that proposed algorithms, both ε-NSGAIII and Adaptive ε-NSGAIII showed better value compared to others. Indicating that their obtained solution is closer to the reference point compared to others.

Table 5.7 Comparative evaluation of Reference Point result on MOTSP

Algorithm	Reference Point		
	Maximum	Average	Minimum
NSGAII	0.084	0.070	0.058
ε-NSGAII	0.077	0.065	0.055
NSGAIII	0.091	0.078	0.062
ε-NSGAIII	0.056	0.052	0.041
A ε-NSGAIII	0.059	0.050	0.038

Since once again, the convergence of the obtained solution cannot differentiate the performance of the proposed algorithm in solving MOTSP, we tried again to evaluate the obtained solutions in terms of the diversity. Spread evaluation can quantify the performance of an algorithm from this point of view.

Table 5.8 Comparative evaluation of Spread Evaluation result on MOTSP

Algorithm	Spread Evaluation		
	Maximum	Average	Minimum
NSGAII	0.941	0.917	0.893
ε-NSGAII	0.935	0.904	0.889
NSGAIII	0.941	0.923	0.907
ε-NSGAIII	0.873	0.851	0.839
A ε-NSGAIII	0.810	0.776	0.744

From the result we can see that obtained solutions from Adaptive ε-NSGAIII has the smallest value among other algorithms, showing that introducing an adaptive concept to ε-NSGAIII could increase the performance of the algorithm in terms of the diversity of obtained solutions.

5.7 Summary

In this chapter, we analyze the performance of Adaptive ε-NSGAIII (A ε-NSGAIII) compared with other algorithms. As the results, although Adaptive ε-NSGAIII has only slightly better performance compares to ε-NSGAIII on a multi-objective TGAP, the algorithm still outperformed other well-known evolutionary algorithms. However, evaluating the result from the diversity point of view revealed that proposed Adaptive ε-NSGAIII was able to overcome the diversity issue in NSGAIII-based algorithm and outperformed other algorithms

In addition, to evaluate the ability of the proposed Adaptive ε-NSGAIII algorithm on other real-life problems, we applied the algorithm on benchmark problem MOTSP. The evaluation of the proposed algorithm on MOTSP was done based on Hypervolume, Generational-Distance, Reference-point Convergence, and Spread Evaluation factors. The results showed that the algorithm could solve MOTSP and in terms of solutions diversity, Adaptive ε-NSGAIII has better performance compared to other algorithms.

Chapter 6 Conclusions

Overall, this dissertation aimed to make and study Tour Guide Assignment Problem (TGAP) modeling with the multi-objective point of view by considering total guiding time, total assignment cost and service quality. Several known multi-objective evolutionary algorithms such as NSGAIII, ε-NSGAII, ε-MOEA, NSGAII, PAES, and PESAII have been used to solve and evaluate the problem in MOEA framework.

Furthermore, the quality of solutions of some well-known evolutionary algorithm has been evaluated on multi-objective TGAP by using performance indicators such as Hypervolume, Generational Distance and Maximum Pareto Front Error. From the results, we concluded that NSGAIII has better performance in terms of running time and number of solutions compared to the other algorithms on multi-objective TGAP.

To increase the performance of the algorithm, we introduced ε-parameter concept into NSGAIII and evaluate the performance by using Hypervolume, Generational Distance and Reference Point Convergence over the benchmark problem ZDT1 and ZDT4. The results showed that introducing the epsilon concept into NSGAIII could increase the performance of the algorithm on a multi-objective TGAP compared to the original NSGAIII and other algorithms.

In addition, we proposed an extended version of ε-NSGAIII named Adaptive ε-NSGAIII by introducing the adaptive ε-parameter concept to improve the algorithm performance. To evaluate its performance, we add another performance indicator named Spread Solution to emphasize the diversity of the obtained solution. As a result, we found that applying adaptive ε-concept could increase the performance of the algorithm on multi-objective TGAP.

Finally, we tested and evaluated the performance of proposed Adaptive ε-NSGAIII on other real-life problems named Multi-objective TSP by using Hypervolume, Generational Distance, Reference Point convergence and Spread Solution performance indicators. The results showed that our proposed algorithm's performance was also able to overcome other well-known MOEAs in obtaining optimum solutions for the problem.

References

[1] Singh, S., Dubey, G. C., and Shrivastava, R., "A Comparative Analysis of Assignment Problem", IOSR Journal of Engineering (IOSRJEN), Volume 2, Issue 8, pp. 1-15, 2012.

[2] Ray, S., Revisiting the Evolution and Application of Assignment Problem: A Brief Overview", Industrial Engineering Letters, Vol.6, No.10, 2016.

[3] A. Wren, "Scheduling, timetabling and rostering – A special relationship? In: Burke and Ross, pp. 46–75, 1996.

[4] Derigs, U., Goecke, O. and Schrader, R., "Monge Sequences and A Simple Assignment Algorithm", Discrete Applied Mathematics, 15, pp. 241-248, 1986.

[5] Chauvet, F. and Proth, J. M., "The Simple and Multiple Job Assignment Problems", International Journal of Production Research, 38(14), pp. 3165-317, 2000.\

[6] Babonneau, F. and Vial, J. P., "An Efficient Method to Compute Traffic Assignment Problems with Elastic Demands", Transportation Science Vol. 42, No. 2, pp. 249-260, 2008.

[7] Garrett, D., Vannucci, J., Silva, R., Dasgupta, D. and Simien, J., "Genetic Algorithms for the Sailor Assignment Problem", GECCO 'O5, June 25–29, Washington, DC, USA, 2005.

[8] Capone, A. and Trubian, M., "Channel Assignment Problem in Cellular Systems: A New Model and a Tabu Search Algorithm", IEEE Transactions on Vehicular Technology, Vol. 48, No. 4, pp. 1252-1260, 1999.

[9] Koopmans, T. C. and Beckmann, M. J., "Assignment problems and the location of economic activities", Econometrica, 25, pp. 53 – 76, 1957.

[10] Cong, J., "Pin assignment with Global Routing for General Cell Design", IEEE Transactions on Computer-aided Design, Vol 10, No 11, pp. 1401-1412, 1991.

[11] Yagiura, M., Ibaraki, T. and Glover, F., "A Path Relinking Approach with Ejection Chains for the Generalized Assignment Problem", European Journal of Operational Research, 169, pp. 548–569, 2006.

[12] Kleeman, M. P. and Lamont, G. B., "The Multi-Objective Constrained Assignment Problem", GECCO'06, July 8–12, Seattle, Washington, USA, pp. 743-744, 2006.

[13] Aardal, K., Van Hoesel, S. P. M., Koster, A. M. C. A., Mannino, C. and Sassano, M., "Models and Solution Techniques for Frequency Assignment Problems", ZIB-Report, Konrad-Zuse-Zentrum fur Informationstechnik Berlin, pp 01–40, 2001.

[14] Chang, C. and Cong, J., “An Efficient Approach to Multilayer Layer Assignment with an Application to Via Minimization”, IEEE Transactions on Computer-Aided Design Of Integrated Circuits and Systems, Vol. 18, No. 5, 1999.

[15] Burke, E. K., Li, J., and Qu, R., “A Pareto-based search methodology for multi-objective nurse scheduling”, Annals of Operations Research, 196(1), pp. 91–109, 2012.

[16] Ji, P., Lee, W. B., and Hongyu, L., “A New Algorithm for the Assignment Problem: An Alternative to the Hungarian Method”, Computers & Operations Research, Volume 24, Issue 11, Pages 1017-1023, 1997.

[17] Machol, R., “Letter to the Editor—An Application of the Assignment Problem”, Operations Research, Vol. 9, No. 4, pp 585-586, 1961.

[18] Chakravarthy, V. K., Ramana, V. V. V. and Umashankar, C., “Investigation of Task Bottleneck Generalized Assignment Problems in Supply Chain Optimization Using Heuristic Techniques, IOSR Journal of Business and Management (IOSR-JBM), Vol. 20, Issue 5, pp. 41-47, 2018.

[19] McGinnis, L. F., “Implementation and Testing of a Primal-Dual Algorithm for the Assignment Problem”, Operations Research, Vol. 31, No. 4, 1983.

[20] Murty, I., and Seo, P. K., “A Nested Dual Ascent Procedure for the Dynamic Single File Allocation Problem”, Information Systems and Operational Research, Volume 31, Issue 3, 1993.

[21] Raghavendra, B.G. and M. Mathirajan, Optimal allocation of buses to depots a case study. OPSEARCH, 24(4), 228-239, 1987.

[22] Ross, G. T. and Zoltmers A. A., “Weighted Assignment Models and their Application”, Management Science, 25, pp. 683-696, 1979.

[23] Elshafei, M. and Alfares, H., A Dynamic Programming Algorithm for Days-off Scheduling with Sequence Dependent Labor Costs, Journal of Scheduling 11(2), pp. 85-93, 2008.

[24] Subrahmanyam, Y. V., “Some special cases of assignment problems”, Opsearch 16 (1), pp. 45-47, 1979.

[25] Volgenant, A., “Linear and Semi-Assignment Problems: A Core Oriented Approach”, Computers & Operations Research Volume 23, Issue 10, pp. 917-932, 1996.

[26] R. Chen, and H. Wang, “Application of Genetic Algorithm to Scheduling of Tour Guides for Tourism and Leisure Industry”, INFOS, Cairo-Egypt, 2008.

[27] Alba, E. and Dorronsoro, B., “Cellular Genetic Algorithms”, ser. Operations Research/ Computer Science Interfaces, Springer-Verlag Heidelberg, 2008.

[28] El Zoghby, M., Badr, A. and Hegazy, A. E. F., “Scheduling of Tour Guides for Tourism and Leisure Industry using Cellular Genetic Algorithm”, IJCSNS, 2011.

[29] Setiyani, L. and Okazaki, T., “3D Neighborhood Relationships of Cellular Genetic Algorithms for the Tour Guide Assignment Problem”, IEIE Transactions on Smart Processing and Computing, vol. 6, no. 3, 2017.

[30] Matthias, E., “Multi-objective optimization”, Association for the Advancement of Artificial Intelligence, 2008.

[31] Nasef, M. M., “Optimization Problems and Optimizations”, Workshop on Intelligent System and Applications, pp. 1-17, 2017.

[32] Zhou, A., Qu, B., Li, H., Zhao, S., Suganthan, P. N. and Zhang, Q., “Multiobjective Evolutionary Algorithms: A survey of the state of the art”. Swarm and Evolutionary Computation 1, 1, pp. 32–49, 2011.

[33] Fonseca, C. M. and Fleming, P. J., “An Overview of Evolutionary Algorithms in Multiobjective Optimization”, Evolutionary Computation, Vol. 3, Issue: 1, 1995.

[34] Li, B., Li, J., Tang, K., and Yao, X., “Many-Objective Evolutionary Algorithms”, ACM Computing Surveys 48(1), 1-35, 2015.

[35] Holland, J. H., “Adaptation in Natural and Artificial Systems”, Ann Arbor: University of Michigan Press.

[36] Civicioglu, F. and Besdok, E., “A Conceptual Comparison of The Cuckoo-Search, Particle Swarm Optimization, Differential Evolution and Artificial Bee Colony Algorithms”, Artificial Intelligence Review, 39 (4), 315-346, 2013.

[37] Hadka, D., and Reed, P., “Borg: An Auto-Adaptive Many-Objective Evolutionary Computing Framework”, Evolutionary Computation, 21 (2), 231-259, 2013.

[38] K. A. De Jong, “Evolutionary Computation: A Unified Approach”. Cambridge, MA, USA: MIT Press, 2006.

[39] K. C. Wong, K. S. Leung, and M. H. Wong, “An Evolutionary Algorithm with Species Specific Explosion for Multimodal Optimization”, GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation, pp. 923-930, New York, NY, USA: ACM, 2009.

[40] Mahdavia, S., Ebrahim, M. and Rahnamayan, S., “Metaheuristics in Large-scale Global Continues Optimization: A survey”, Information Sciences, Vol 295, 407-428, 2015.

[41] Goldberg, D. E., “Genetic algorithms in search, optimization and machine learning”, Addison-Wesley, 1989.

[42] Mausser, H., “Normalization and Other Topics in Multi-Objective Optimization”,

Proceedings of the Fields–MITACS Industrial Problems Workshop, pp. 89-101, 2006.

[43] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II", IEEE Transaction on Evolutionary Computation, vol. 6, no. 2, pp. 182–197, 2000.

[44] K. Deb, M. Mohan, and S. Mishra, "A Fast Multi-Objective Evolutionary Algorithm For Finding Well-Spread Pareto-Optimal Solutions," KanGAL Report 2003002, January 2003.

[45] Deb. K., Jain, H., "An Evolutionary Many-Objective Optimization Algorithm Using Reference-Point-Based Nondominated Sorting Approach, Part I: Solving Problems with Box Constraints", IEEE Transactions on Evolutionary Computation, vol. 18, issue: 4, pages: 577 – 601, 2014.

[46] Rajnikant, H., Bhesdadiya, R.H., Trivedi, I. N., Jangir, P., Jangir, N., Kumar, A., "An NSGA-III Algorithm for Solving Multi-Objective Economic/Environmental Dispatch Problem", Cogent Engineering, 3: 1269383, 2016.

[47] J. Knowles and D. W. Corne, "Approximating the Nondominated Front Using the Pareto Archived Evolution Strategy," Evolutionary Computation, vol. 8, no. 2, pp. 149–172, 2000.

[48] D. W. Corne, J. D. Knowles, and M. J. Oates, "The Pareto Envelope-Based Selection Algorithm for Multiobjective Optimization," Proceedings of the Parallel Problem Solving from Nature VI Conference, pp. 839–848, 2000.

[49] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the Strength Pareto Evolutionary Algorithm," In: Evolutionary Methods for Design, Optimization, And Control, pp. 95–100, 2002.

[50] E. Zitzler, K. Deb and L. Thiele, "Comparison of Multi-objective Evolutionary Algorithms: Empirical Results", Evolutionary Computation archive, Vol 8 Issue 2, 173-195, 2000.

[51] M. Laumanns, L. Thiele, K. Deb and E. Zitzler, Combining Convergence and Diversity in Evolutionary Multi-objective Optimization, Evolutionary Computation, 10(3), 2002.

[52] Kursawe, F., "A variant of evolution strategies for vector optimization" In H.-P. Schwefel and R. M¨anner (Eds.), Parallel Problem Solving from Nature, Springer, Berlin, pp. 193–197, 1991.

[53] Liao, Q., Sheng, Z., Shi, H., Zhang L., Zhou, L., Ge, W. and Long, Z., "A Comparative Study on Evolutionary Multi-objective Optimization Algorithms Estimating Surface Duct", Sensors, 18(12), pp. 4428, 2018.

[54] E. Zitzler and L. Thiele, “An Evolutionary Algorithm for Multi-objective Optimization: The Strength Pareto Approach”, Evolutionary Computation archive, Vol 8 Issue 2, 173-195, 2000.

[55] Kollat, J. B. and Reed, P. M., Comparing State-of-the-Art Evolutionary Multi-Objective Algorithms for Long-Term Groundwater Monitoring Design, Advances in Water Resources, Volume 29, Issue 6, pp. 792-807, 2006.

[56] Mane, S. and Rao, M. R. N., “Many-Objective Optimization: Problems and Evolutionary Algorithms–A Short Review”, International Journal of Applied Engineering Research 12(20), 2017.

[57] I. C. Parmee, “Evolutionary and Adaptive Computing in Engineering Design”, Springer, pp 200-202, 2012.

[58] G. Gutin, A. Punnen, “The Traveling Salesman Problem and Its Variations”, Kluwer Academic Publishers, Dordrecht, 2002.

[59] N. Qamar, N. Akhtar, I. Younas, “Comparative Analysis of Evolutionary Algorithms for Multi-Objective Travelling Salesman Problem”, International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 9, No. 2, 2018.

[60] Ghoreishi, S. N., Sørensen, J. C., Jørgensen, B. N., “Comparative Study of Evolutionary Multi-Objective Optimization Algorithms for A Non-Linear Greenhouse Climate Control Problem”, IEEE Congress on Evolutionary Computation (CEC), 2015.

[61] Deb, K., Sundar, J., N., U. B. R. & Chaudhuri, S.,”Reference Point Based Multi-Objective Optimization Using Evolutionary Algorithms”, International Journal of Computational Intelligence Research, ISSN 0973-1873, 2(3), p. 273–286, 2006.

[62] F. Siegmund, Ng, A. H. and K. Deb, “Finding A Preferred Diverse Set of Pareto-Optimal Solutions for A Limited Number of Function Calls”, Brisbane, Australia, WCCI 2012 IEEE World Congress on Computational Intelligence, 2012.