An Improved Multi-Objective Evolutionary Algorithm with Adaptable Parameters

by

Khoa Duc Tran

A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in
Computer Information Systems

Graduate School of Computer and Information Sciences
Nova Southeastern University

2006

We hereby certify that this dissertation, submitted by Khoa Duc Tran, conforms to acceptable standards and is fully adequate in scope and quality to fulfill the dissertation requirements for the degree of Doctor of Philosophy.

Sumitra Mukherjee, Ph.D. Date
Chairperson of Dissertation Committee

Michael J. Laszlo, Ph.D. Date
Dissertation Committee Member

Easwar A. Nyshadham, Ph.D. Date
Dissertation Committee Member

Approved:

Edward Lieblein, Ph.D. Date
Dean

Graduate School of Computer and Information Sciences
Nova Southeastern University

2006

An Abstract of a Dissertation Submitted to Nova Southeastern University in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

An Improved Multi-Objective Evolutionary Algorithm with Adaptable Parameters

by
Khoa Duc Tran

August 2006

Multi-Objective Evolutionary Algorithms (MOEAs) are not easy to use because they require parameter tunings of three main parameters - population size, crossover probability, and mutation probability - in order to achieve desirable solutions and performance for an arbitrary complex problem. Moreover, the use of fixed parameter settings may lead to slow convergence and sub-optimal solutions. This dissertation develops a MOEA with self-adaptive crossover, self-adaptive mutation, and adaptive population size parameters for automating the process of adjusting appropriate parameter values in order to make the MOEA more efficient, simple to use and available to more users. The MOEA with adaptable parameters is built on the NSGA-II (Non-dominated Sorting Genetic Algorithm II) and named as ANSGA-II (Adaptable NSGA-II). The NSGA-II is chosen because it is one of the best-known MOEAs. In the ANSGA-II, the crossover and mutation parameters are attached to each solution in the population and allowed to co-evolve with each solution. This enables the algorithm to carry prior successful crossover and mutation for creating children solutions and for adaptation of the parameters. Since good parameter values are associated with good candidate solutions, better parameter values will survive because they produce better solutions. The ANSGA-II selects the right population size by running several populations with different population sizes simultaneously and allows the smaller populations more time to run. Smaller populations may find diverse non-dominated solution sets close to the Pareto-optimal front faster than the larger populations. If a subsequent larger population identifies a better non-dominated solution set then the algorithm stops running the smaller population since it is unlikely to identify better solutions than the larger one due to genetic drift. Two performance metrics are investigated for their effective use in comparing non-dominated solution sets among different populations during the execution of the ANSGA-II. The dissertation evaluates and discusses the performance of the ANSGA-II, in terms of finding a diverse non-dominated solution set and converging to the true Pareto-optimal front, by comparing the results obtained on a suite of thirteen benchmark multi-objective problems with those obtained by the original NSGA-II. The results demonstrate that the ANSGA-II out-performs the NSGA-II. The improvement comes with the cost of longer execution time due to overheads of finding good non-dominated solutions and learning good parameter values at the same time. However, the execution time appears to be acceptable on all thirteen benchmark multi-objective problems.

Acknowledgments

I would like to thank several people for their helps and supports during my dissertation work.

Dr. Sumitra Mukherjee for his willingness to be the chair of my dissertation committee, his valuable comments and suggestions in producing a quality dissertation report.

Dr. Michael J. Laszlo and Dr. Easwar A. Nyshadham for accepting to be the committee member of my dissertation, and their valuable comments.

Dr. Kalyanmoy Deb and Dr. Fernado G. Lobo for having their source code available online.

I would also like to thank God, my family, and my friends. Without their supports and encouragements, I could not have completed my PhD.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

This chapter presents an introduction to this dissertation. It is organized as follows: First, it describes the problem to be investigated and goal to be achieved. Second, it provides the relevance and significance for the research. Third, it describes barriers and issues that have prevented the goal to be achieved. Fourth, it continues by presenting research questions that the dissertation investigates and provides the answers. Fifth, it describes the limitations and delimitations of this study. Sixth, it provides definitions of key terms used throughout this study. Finally, it presents a summary of this chapter.

Problem Statement and Goal

This section provides a brief background of Multi-Objective Evolutionary Algorithms (MOEAs). Thereafter, it presents the problem statement and goal of the dissertation.

Background

Real-world optimization problems often involve simultaneous optimization of multiple and conflicting objectives. In a multi-objective optimization problem (MOP), it is not always possible to find a solution that is best with respect to all objectives. A solution may be optimal regarding one objective, but at the same time be inferior regarding another objective. Typically, the goal is to find a set of optimal trade-off solutions known as Pareto-optimal set. These solutions are optimal in a broader sense that no other solutions in the search space are better to them when all objectives are considered. Since none of the solution in the set is absolutely better than any other solution with respect to all objectives, any one of them is an acceptable solution. The choice of one solution over the other requires problem knowledge and a number of problem-related factors, often taken by a decision maker (human) (Osyczka, 1985; Deb, 2001; Dias & Vasconcelos, 2002). Some basic concepts used in multi-objective optimization are presented in the following.

Definition of a Multi-Objective Problem (MOP): Formally, a MOP can be defined as follows:

Find the vector $\boldsymbol{x}^* = \left[x_1^*, x_2^*, ..., x_n^*\right]$, which minimize (or maximize, since min[$f(\boldsymbol{x})$] = -max[-$f(\boldsymbol{x})$]) the vector function:

$$f(\boldsymbol{x}) = \left[f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), ..., f_m(\boldsymbol{x})\right] \in Z \tag{1}$$

and will satisfy the p inequality constraints:

$$g_i(\boldsymbol{x}) \geq 0 \quad i = 1, 2, ..., p \tag{2}$$

and the q equality constraints:

$$h_i(\boldsymbol{x}) = 0 \quad i = 1, 2, ..., q \tag{3}$$

where m is the number of objective functions, $\boldsymbol{x} = \left[x_1, x_2, ..., x_n\right] \in X$ is the vector of decision variables, $X \in R^n$ is the n-dimensional decision space, and $Z \in R^m$ is the m-dimensional objective space. The function f: $X \rightarrow Z$ evaluates the quality of a specific solution by assigning it an objective vector [z_1, z_2,..., z_m] $\in Z$. Thus, for each solution $\boldsymbol{x}$ in the decision space, there exists a corresponding solution in the objective space, denoted by $f(\boldsymbol{x}) = \boldsymbol{z} = \left[z_1, z_2, ..., z_m\right] \in Z$. Fonseca and Fleming (1995) pointed out that when there is no priori preference defined among the objectives, dominance is the only way to determine if one solution is better than the other solution. Figure 1 illustrates a general MOP.

Dominance Relation: A solution $x_a \in X$ is said to dominate a solution $x_b \in X$ (denoted as $x_a \prec x_b$) if it is better than or equal to x_b in all objectives (i.e. $f_i(x_a) \le f_i(x_b)$ for all $i = 1,\ldots,m$) and at least better than x_b in one objective (i.e. $f_j(x_a) < f_j(x_b)$ for at least one $j = 1,\ldots,m$).

Non-dominated: A solution is referred as non-dominated if it is not dominated by any other solutions.

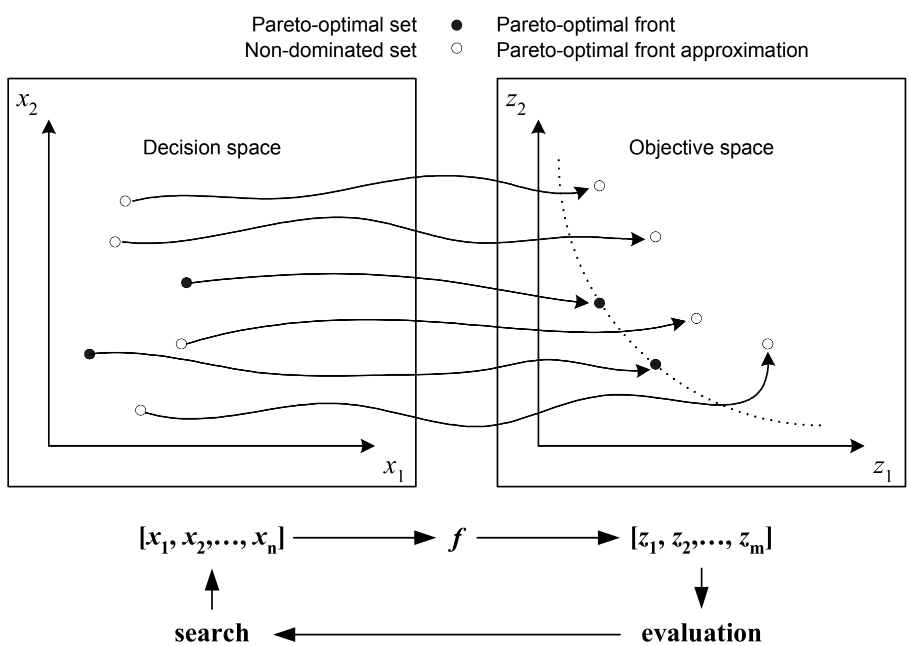


Figure 1: Illustration of a general multi-objective optimization problem

Goal of Multi-Objective Optimization: The goal of multi-objective optimization is to find among the set X of all vectors, which satisfy (2) and (3), the particular set $x_1^*, x_2^*, \ldots, x_n^*$ which yields the optimal values of all objective functions.

Pareto optimal: A vector of decision variables $\boldsymbol{x}^* \in X$ is Pareto optimal if $\boldsymbol{x}^*$ is non-dominated with respect to the set of all possible vectors in X (i.e., if there does not exist another $\boldsymbol{x} \in X$ such that $f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{x}^*)$ for all i = 1,…, m and $f_j(\boldsymbol{x}) < f_j(\boldsymbol{x}^*)$ for at least one j = 1,…,m). Pareto-optimal vectors or Pareto-optimal solutions are characterized by the fact that improvement in any one objective means worsening at least one other objective.

Pareto-optimal Set: The Pareto-optimal set (denoted as P^*) is the set of all possible Pareto-optimal solutions (i.e., $P^* = \left\{\boldsymbol{x}^* \in X \mid \neg\exists\, \boldsymbol{x} \in X \text{ and } f(\boldsymbol{x}) \prec f(\boldsymbol{x}^*)\right\}$). A Pareto-optimal set is always a non-dominated set. However, a non-dominated set may contains some Pareto-optimal solutions and some non-Pareto-optimal solutions.

Pareto-optimal Front: The plot of the objective functions whose non-dominated solutions are in the Pareto-optimal set is called the Pareto-optimal front (denoted as PF^*) (i.e. $PF^* = \left\{f(\boldsymbol{x}^*) = \left[f_1(\boldsymbol{x}^*), f_2(\boldsymbol{x}^*), \ldots, f_m(\boldsymbol{x}^*)\right] \mid \boldsymbol{x}^* \in P^*\right\}$). Figure 1 above and Figure 2 below illustrate Pareto-optimal solutions and Pareto-optimal front.

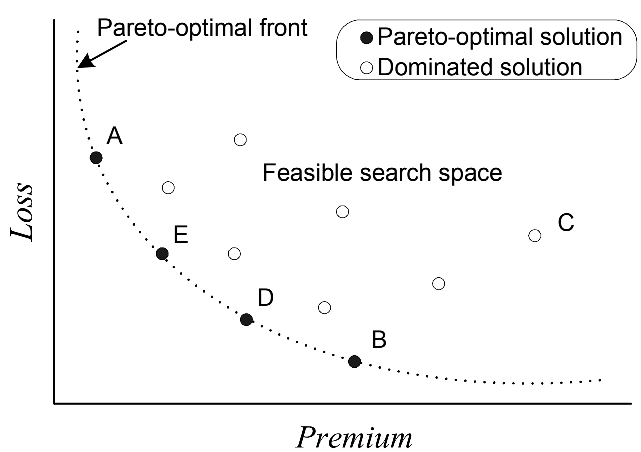


Figure 2: Illustration of Pareto-optimal solutions and Pareto-optimal front

Figure 2 provides a hypothetical example of purchasing a car insurance policy. The purchaser wishes to reduce monetary loss in case of a car accident but at the same time he wishes to pay low premium on the policy. Thus, there are two conflicting objectives, loss and premium, both of which are to be minimized. The purchaser also wishes the policy to have the following coverages or conditions: collision and comprehensive coverages. In mathematical terminology, the available limits and deductibles are the problem's decision variables, the conditions to be met are the constraints, and the process of minimizing and maximizing the objectives is called optimization. In Figure 2, the point A represents a solution, which has a near minimum premium but high loss. On the other hand, the point B represents a solution, which has a high premium but near least loss. If both objectives are considered, one cannot conclude if solution A is better than solution B, or vice versa. One solution is better than other in one objective, but is worse in the other. In Figure 2, the Pareto-optimal front is marked by the dashed line. All the trade-off solutions along the Pareto-optimal front are known as Pareto-optimal solutions. The area behind the Pareto-optimal front is known as feasible search space or feasible objective space. In front of Pareto-optimal front is infeasible search space, in which solutions are un-attainable corresponding to the optimality of both objectives. There exist non-Pareto-optimal solutions such as the point C. When the solution C is compared with the solution A, again one cannot conclude whether one is better than the other in both objectives. However, the solution C is not a member of the Pareto-optimal set because there exists another solution D in the search space, which is better than the solution C in both objectives. The solutions such as C are known as dominated solutions or inferior solutions.

Goals of a Multi-Objective Optimization Algorithm: The goal of a multi-objective optimization algorithm is to (i) guide the search towards the global Pareto-optimal front and to (ii) maintain solution diversity in the Pareto-optimal front (Deb, 2001). These two goals are distinct and in some sense orthogonal to each other as illustrated in Figure 3. The non-dominated solutions found by a multi-objective optimization algorithm might not represent the true Pareto-optimal set but approximate the true Pareto-optimal set.

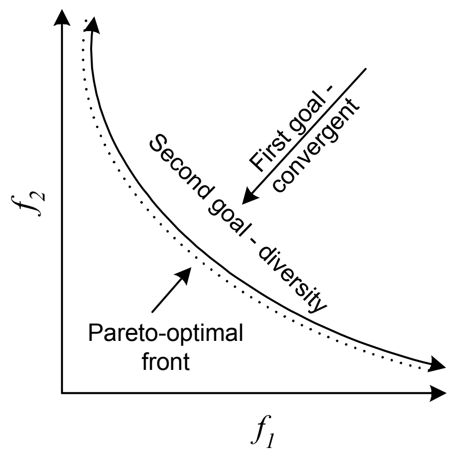


Figure 3: Two goals of a multi-objective optimization algorithm

A Multi-Objective Evolutionary Algorithm (MOEA) is a modified version of the traditional Genetic Algorithm (GA) - also known as the simple GA, designed to solve multi-objective optimization problems (MOPs). MOEAs have been recognized to be suitable for solving multi-objective optimization problems because of their ability to find good solutions for competing objective functions simultaneously and to search for multiple non-dominated solutions simultaneously in the population of candidate solutions. This capability enables them to find several trade-off solutions for all objectives in a single run of the algorithm, instead of having to perform a series of separate runs as in the case of the traditional techniques such as weighted sum, goal programming and weighted min-max methods (Khare, Yao, & Deb, 2002; Büche, Müller,

& Koumoutsakos, 2003). Ideally, a MOEA returns a Pareto-optimal set, the solutions not dominated by any other solution in the search space. In addition, MOEAs are less susceptible to the shape or continuity of the Pareto-optimal front, whereas these two issues pose a problem for the mentioned traditional techniques.

Problem Statement

As with the simple GAs, MOEAs are not easy to use because they require parameter tunings of population size, crossover probability, and mutation probability in order to achieve the desirable solutions and performance for an arbitrary complex problem. The task of tuning GA parameters is not trivial due to the complex and nonlinear interactions among the parameters and their dependency on many aspects of the particular problem being solved such as the search space size and the shape of the fitness surface (Deb & Agrawal, 1998; Eiben, Hinterding, & Michalewicz, 1999; Mitchell, 2002). As a result, these GA parameters cannot be optimized one at a time; and trying all different combinations systematically is practically impossible (Eiben et al., 1999). Another problem is that the proper parameter values are not fixed but varied during a run because a GA is dynamic and adaptive process (Fogarty, 1989; Davis, 1991; Hesser & Männer, 1991; Bäck, 1992; Mühlenbein, 1992; Laumanns, Rudolph, & Schwefel, 2001). Therefore, the use of fixed parameter settings may lead to slow convergence and sub-optimal obtained solutions (i.e. solutions are not close to the Pareto-optimal front), especially when large search spaces are to be explored in solving complex optimization problems. A possible solution to these problems is to monitor a GA's progress in order to adjust its parameter values during its execution. This technique is known as parameter control. Eiben et al. (1999) classified parameter control into three different categories:

deterministic control, adaptive control, and self-adaptive control. In deterministic control, the parameter values are changed during a run according to some deterministic rule, which usually depends on time such as number of generations. In adaptive control, information fed back from the GA, during its run, is used to adjust the parameter values and all individuals in the population have the same parameter values. In self-adaptive control, information fed back from the GA, during its execution, is used to adjust the values of parameters attached to each individual in the population. Thus, the parameters are co-evolved with each individual in the population and each individual in the population can have different parameter values in the self-adaptive control technique. Although previous studies have applied parameter control techniques to MOEAs (Laumanns et al., 2001; Sbalzarini, Müller, & Koumoutsakos, 2001; H.A. Abbass, 2002; Büche et al., 2003; Devireddy & Reed, 2004), most of these studies focus on one or two parameters in isolation and ignore other parameters. This dissertation aims to investigate simultaneous parameter control techniques in MOEA for all three parameters - population size, crossover, and mutation.

Dissertation Goal

The objective of this dissertation is to develop a MOEA with self-adaptive crossover, self-adaptive mutation, and adaptive population size parameters for automating the process of selecting appropriate parameter values. This MOEA is built on the NSGA-II (Non-dominated Sorting Genetic Algorithm II) (Deb, Pratap, Agarwal, & Meyarivan, 2002) and named as ANSGA-II (Adaptable NSGA-II). The NSGA-II, which supports static parameters, is selected in this study because it has been recognized to perform as well or better than other MOEAs with the same goal of finding a diverse

Pareto-optimal solution set such as the Pareto-Archived Evolution Strategy (PAES) and Strength Pareto Evolutionary Algorithm 2 (SPEA2) (Zitzler, Laumanns, & Thiele, 2002; Devireddy & Reed, 2004).

In the ANSGA-II, the crossover and mutation parameters are attached to each solution in the population and allowed to co-evolve with each solution. This enables the algorithm to carry prior successful crossover and mutation for creating children solutions and for adaptation of these two parameters since good crossover and mutation probabilities are associated with good candidate solutions. The ANSGA-II, adopting the multiple population approach of Harik and Lobo (1999), runs several populations with different population sizes simultaneously. Two running performance metrics are integrated into the ANSGA-II and investigated for their effective use in comparing non-dominated solution sets among different populations during the execution of the ANSGA-II – one for measuring the convergence to a reference set suggested by Deb & Jain (2002) and other for measuring the diversity of solutions suggested by Deb (2002). The convergent metric of Deb and Jain, which measures the closeness of solutions to the Pareto-optimal front is evaluated in this study because it can work with an unknown set of Pareto-optimal solutions by using a population-agglomeration technique and it is computational fast (Deb & Jain, 2002). The diversity metric, which measures the spread of the obtained solutions, is adopted in this study because it was used in the original study of NSGA-II (Deb, Pratap et al., 2002) for measuring the diversity of the obtained solutions. These two metrics can provide a comparative evaluation of two or more MOEAs, or compare two or more non-dominated solution sets. They can also enable the ANSGA-II to determine when the Pareto-optimal solutions have been sufficiently obtained by calculating the change in the metric results. If the differences of these two metric results fall within a certain threshold (e.g. 0.01) respectively for two successive runs then the obtained non-dominated solutions can be considered identical and the algorithm can terminate with a proper population size. The dissertation then evaluates and discusses the performance of the ANSGA-II, in term of finding a diverse Pareto-optimal solution set, by comparing the results obtained on the benchmark multi-objective test problems with those obtained by the original NSGA-II. These test problems, which have been used to evaluate the NSGA-II (Deb, Pratap et al., 2002), consist of nine un-constrained test problems with two objective functions, three constrained test problems with two objective functions, and one real-world problem with five objective functions and seven constraints (see Appendix A).

Relevance and Significance

MOEAs have been used increasingly in a wide range of real-world multi-objective optimization applications including but not limited to: telecommunication network design (Flores, Cegla, & Cáceres, 2003; Maple, Guo, & Zhang, 2004), software quality enhancement (Khoshgoftar, 2004), risk-based corrective action design (Gopalakrishnan, Minsker, & Padera, 2001), optimization of corrugated bulkhead forms (Yang & Hwang, 2002), digital filter design (Schnier, Yao, & Liu, 2001). Today, the MOEA repository (http://www.lania.mx/~ccoello/EMOO/) contains over 2178 papers, from which a vast majority are applications (Coello, 2005).

According to Eiben et al. (1999) and Laumanns et al. (2001), the issue of controlling values of various parameters of a GA is one of the most important and promising areas of research in evolutionary computation. It has a potential of adjusting the algorithm to the problem while solving the problem and the user does not have to specify or tune the values of the parameters. Several parameter control methods have been proposed and applied successfully for single objective optimization problems using simple GAs. Both theoretical and empirical studies have suggested that the proper mutation probability (a proper mutation probability enables diversity in the population without destroying already found good solutions) varies with evolutionary time according to the state of the search and the nature of the search problem (Fogarty, 1989; Hesser & Männer, 1991; Bäck, 1992; Mühlenbein, 1992). Davis (1991) applied a time-varying schedule of parameter settings and found that performance was improved. Spears (1991) applied self-adaptation for selecting optimal crossover operator (uniform crossover or two-point crossover) and showed that this adaptive crossover operator out-performs non-adaptive crossover operator, especially with large population sizes. Smith & Fogarty (1996) used self-adaptation for mutation rate and showed that the self-adaptive mutation significantly improves the GA's performance as well as making it possible to remove the mutation parameter from the set of decisions faced by the user. However, research focusing on the role of parameter control in MOEAs remains rare. Most MOEAs such as NSGA-II, PAES, and SPEA2 support static parameters, where the parameter settings are initialized at the beginning of a MOEA's execution and fixed during the course of its execution. The use of fixed parameter settings may lead to slow convergence and sub-optimal obtained solutions (i.e. solutions are not well spread and not close to the Pareto-optimal front), especially when large search spaces are to be explored in solving complex optimization problems because the proper parameter values are not fixed but varied during a run and a MOEA is dynamic and adaptive process. Although some previous studies have applied parameter control techniques to MOEAs (Laumanns et al., 2001; H.A. Abbass, 2002; Büche et al., 2003; Devireddy & Reed, 2004), these studies focus on one or two parameters in isolation and ignore other parameters. Therefore, more research needs to be done on parameter control techniques for MOEAs.

This dissertation aims to investigate simultaneous parameter control techniques in MOEA for all three parameters – population size, crossover, and mutation. Hence, it significantly advance knowledge in the research area of parameter control techniques for MOEAs and improve MOEAs by making them more efficient, simple to use and available to more users.

Barriers and Issues

A MOEA with simultaneous parameter control technique for all three parameters population size, crossover, and mutation has not been developed for a number of reasons. One reason is that applying self-adaptive crossover and mutation to individuals reveals an additional difficulty compared to single objective optimization. Self-adaptive parameters benefit from the recombination of many parent solutions. However, the result of the multi-objective optimization process is usually not a single solutions but a set of trade-off solutions. These trade-off solutions converge towards different areas of the Pareto-optimal front and proper parameter values differ between these solutions (Büche et al., 2003). Büche et al. also pointed out that the dominance criterion based MOEAs work well for approximation of the Pareto front, but fail in the final convergence since the archive size of this class of algorithms is usually limited. Deb (2001) also agreed with this point as described in the next paragraph.

Deb (2001) pointed out two disadvantages of the NSGA-II. The first disadvantage is that Pareto-optimal solutions may be replaced by other inferior non-dominated solutions due to the way the algorithm preserve elitism. Elitism in NSGA-II is ensured by comparing the current population with previously found best non-dominated solutions and by combining the parent and child populations to form a combined population with size 2*N* (Figure 8 below illustrates this process). The combined population is then sorted according to non-domination. Solutions belonging to the first non-dominated front are of the best solutions in the combined population. As long as the size of the first non-dominated set is not larger than the population size, the algorithm preserves all of them in the new population of size *N*. However, in later generation, when the first non-dominated set has nearly converged to the Pareto-optimal set, there might be more than *N* solutions in the first non-dominated set of the combined parent-offspring population, only those solutions with greater crowding distance (less crowded area) are chosen. In doing so, the algorithm has no way to know which solutions are already Pareto-optimal and which are not Pareto-optimal (but non-dominated). As a result, already found Pareto-optimal solutions may be replaced by other inferior non-dominated solutions and convergence cannot be guaranteed. Although, in a later generation these replacing non-dominated solutions may get dominated by other Pareto-optimal solutions, the algorithm can go into the cycle of generating Pareto-optimal and non-Pareto-optimal solutions before finally converging to a wide spread set of Pareto-optimal solutions. Thus, the algorithm wastes computational resources. The second disadvantage is that the elitism requires a population of size 2*N*, instead of size *N* required in most other MOEAs.

Tran (2005) integrated the parameter-less GA approach (Harik & Lobo, 1999) into the NSGA-II and named the modified version as parameter-less NSGA-II (Chapter 2 – Review of Literature provides descriptions of the parameter-less GA and NSGA-II). As in the parameter-less GA and the NSGA-II, the parameter-less NSGA-II has the following parameter settings internally within the algorithm:

- Ignores mutation by setting $p_m = 0$;
- Crossover probability $p_c = 0.5$;
- Distribution indices for real-coded crossover operator $\eta_c = 20;$
- Adaptive population size: By establishing a race among multiple populations with different population sizes. Each population is allowed to run up to a maximum number of generations equals to 500.

The result of this study shows that this adaptive population size with multiple population approach does not work well for a Pareto-based MOEA like the parameter-less NSGA-II. In the parameter-less GA (single-objective case), each solution in the population is evaluated by using a fitness function (often the same as the objective function) and assigned an absolute fitness value. In order to determine the proper population size, solutions in two populations are compared for better fitness. For example, if a smaller population has not converged but it has the average fitness better than that of the larger population then there is no need to continue running the larger population because the smaller population size is the proper one. In contrast to single-objective optimization, in multi-objective optimization, both fitness assignment and selection must support several objectives. In a MOEA, each solution is assigned a fitness value equal to its non-dominated rank in the population (1 is the best rank, 2 is the next best rank and so on), which is determined by using a non-dominated sorting procedure (as described in Chapter 2 – Review of Literature below). Solutions that have the best rank values (rank 1) constitute the first non-dominated front. In order to determine the proper population size, solutions in two populations are compared for better non-dominated solutions, which means that the solutions in the first non-dominated front of a smaller population are compared with the solutions in the first non-dominated front of a larger population. However, all of these solutions have the same rank values. As a result, there is no simple way to determine the better non-dominated solution set between two non-dominated solution sets in different populations and the parameter-less NSGA-II fails to determine a proper population size. Performance metrics can be integrated into a MOEA to measure the quality of the obtained solutions during its run in order to monitor and provide progress information for adjusting the values of parameters. However, most of the existing performance metrics available in published literature are applied to the final non-dominated set obtained by a MOEA to evaluate its performance and may not be computationally efficient to be used as running performance metrics (Deb & Jain, 2002). Until recently, studies have introduced some performance metrics that are suitable to be used as running performance metrics (H.A. Abbass, 2002; Deb & Jain, 2002; Farhang-Mehr & Azarm, 2002; Lu & Yen, 2002).

Mitchell (2002) pointed out that a big issue for any adaptive parameter approach is that how to match the rate of adaptation for parameter settings with the adaptation rate of solutions in the GA population. This issue is also applicable for a MOEA. The feedback from the running MOEA (i.e. obtained from the running metrics) is used to adjust the values of the parameters, but it might be difficult and computationally expensive to keep this feedback information current enough for the parameter settings to catch up with the population's current state. According to Mitchell, very little research has been done on measuring these different rates of adaptation and how well they match in different adaptive parameter approaches.

Despite the fact that many new and improved MOEAs have been introduced, there severely lack for studies related to theoretical convergence analysis with guaranteed spread of solutions in MOEAs (Deb, 2001; Laumanns, Thiele, Deb, & Zitzler, 2002). In this regard, several studies have proposed a number of MOEAs, which ensure convergence to the true Pareto-optimal set (Rudolph, 1998; Veldhuizen & Lamont, 1998; Hanne, 2000b, 2000a; Rudolph & Agapie, 2000; Rudolph, 2001). However, these MOEAs do not guarantee the diversity of the obtained non-dominated set. In the ideal approach to multi-objective optimization, there are two tasks: minimize the distance of the obtained solutions to the Pareto-optimal set and maximize the diversity of the obtained non-dominated set. Since achievement of one task does not automatically guarantee achievement of the other task. Thus, in addition to a proof for convergence to the true Pareto-optimal set, it is also necessary to have a proof of diversity of the obtained non-dominated set. Until recently, Laumanns et al. (2002) proposed a new class of MOEAs based on the ε-dominance concept (see *ε-Dominance Relation* in section Definition of Terms) which have both properties of convergence to the true Pareto-optimal set and diversity of the obtained non-dominated set together. They also provided a proof of convergence to the true Pareto-optimal set while preserving diversity of the obtained solutions at the same time.

Research Questions

Accomplishment of this dissertation answers the following research questions:

- Do adaptable parameters (population size, crossover, mutation) improve the performance of the ANSGA-II in term of finding a diverse set of Pareto-optimal solutions?
- Are the convergent and diversity metrics reliable for measuring the quality of the obtained solution set in term of approximating to the true Pareto-optimal set and diversity of solutions?
- Can the running convergent and diversity metrics be used effectively for comparing two or more non-dominated solution sets during the execution of the ANSGA-II?
- Can the running convergent and diversity metrics used in this study be applied for solving problems with more than two objectives?
- Does the ANSGA-II increase or decrease the number of function evaluations compared to the original NSGA-II for solving the same test problems?
- Is the overhead for learning good parameter values acceptable in the ANSGA-II?
- Which adaptable parameter among three adaptable parameters affects the performance of the ANSGA-II the most?

- Which adaptable parameter among three adaptable parameters affects the performance of the ANSGA-II the least?
- Does the combination of all three adaptable parameters work better?
- What is the effect of using each adaptable parameter separately (i.e. ANSGA-II with self-adaptive mutation only, ANSGA-II with self-adaptive crossover only, and ANSGA-II with adaptive population size only)?
- What is the effect of using the combination of two adaptable parameters (i.e. ANSGA-II with adaptive population size and self-adaptive crossover, ANSGA-II with adaptive population size and self-adaptive mutation, and ANSGA-II with self-adaptive crossover and self-adaptive mutation)?

Limitations and Delimitations

The ANSGA-II relies on the convergent and diversity metrics to monitor the progress of the algorithm during its run in order to adjust the values of its parameter accordingly. Several performance metrics have been introduced in the MOEA literature. However, most of these metrics are applicable to two-objective problems (Deb & Jain, 2002). The convergent metric and diversity metric investigated in this study also have not been applied to measure the quality of solutions on problems with more than three objectives. For example, in the original study of NSGA-II, Deb et al. (2002) did not provide convergent and diversity metric results on the five-objective WATER problem (Appendix A provides the listing of this problem). Instead, they provided the lower and upper bounds of each objective function values on the obtained non-dominated solutions for this problem. Since, development of better performance metrics is beyond the scope of this dissertation, existing performance metrics are investigated for their effective use in comparing non-dominated solution sets among different populations during the execution of the ANSGA-II in order to support adaptable population size.

This dissertation narrows its focus on the Pareto-based MOEAs (see Definition of Terms in this chapter) mainly because until today most of the successful MOEAs are Pareto-based approaches derived from the non-dominated sorting procedure (Deb, 2003; Coello, 2005).

This dissertation evaluates the performance of the ANSGA-II, in term of finding a diverse Pareto-optimal solution set, by comparing the results obtained on thirteen multi-objective test problems with those obtained by the original NSGA-II. The NSGA-II serves as a benchmark the ANSGA-II. However, this dissertation does not provide a proof of convergence to the true Pareto-optimal set and diversity of the obtained solutions because there severely lack for studies related to theoretical convergence analysis with guaranteed spread of solutions in MOEAs (Deb, 2001; Laumanns et al., 2002) and development of theoretical proofs is beyond the scope of this dissertation.

Definition of Terms

The following are definitions of key terms used throughout this study.

Adaptive Parameter Control: is a parameter control technique that adjusts the parameter values according to information fed back from the GA during its run. All individuals in the population have the same parameter values.

Aggregating Functions: are traditional approaches for solving multi-objective problems by combining multiple objectives into a single-objective.

Algorithm: an algorithm is a finite set of well-defined instructions for accomplishing some task.

Best-First Search: is a search algorithm, which optimizes depth-first search by expanding the most promising node chosen according to some rule.

Binary-Coded GA: a GA that is based on binary-coded representation.

Binary-Coded Representation: Each gene represents a decision variable of the problem and coded as a binary string (strings of ones and zeros, or bits). The binary coding of all the problem's decision variables can be concatenated to form a fixed-length binary string (chromosome). For example, the following binary string represents a chromosome of four variables:

$$\underset{x_1}{10011}\ \underset{x_2}{01010111}\ \underset{x_3}{101}\ \underset{x_4}{1010}$$

Breadth-First Search: is an algorithm for traversing or searching a tree or graph. It is an uninformed search that progresses by traversing the tree or graph in a level-by-level fashion. The algorithm starts at the root node and explores all the neighboring nodes. Then for each of those nearest nodes, it explores their unexplored neighbor nodes, and so on, until it finds the goal.

Building Blocks (BBs): are schemas (see Definition of Terms) that have short defining lengths and above-average fitness. For example, the schema 01*** has only two defined bits and they are close to each other. If the string 01000 represents the optimal solution for a problem then the schema 01*** has above-average fitness. Hence the schema 01*** is a building block. Schemas with short defining lengths are preferred because crossover is disruptive, the longer the defining length of a schema, the higher chance that the crossover point will fall between its fixed positions and an instance will be destroyed.

Chromosome: or individual refers to a candidate solution to a problem (in binary-coded representation a chromosome is encoded as a fixed-length binary string).

Convergent Metric: In MOEA, a convergent metric is used to measure the closeness of a solution set to the Pareto-optimal front or the true Pareto-optimal set.

Crossover or Recombination: takes two parental individuals, swapping components to produce two offspring that are likely to be better individuals. There exist various crossover operators such as multi-point crossover (Spears & De Jong, 1991), order crossover (Davis, 1985; Oliver, Smith, & Holland, 1987), and others (Booker, Fogel, Whitley, & Angeline, 1997; Spears, 1997). Figure 4 illustrates three commonly used crossover operators: one-point crossover, two-point crossover, and uniform crossover. These crossover operators are also classified as fixed crossover operators because their crossover sites are fixed.

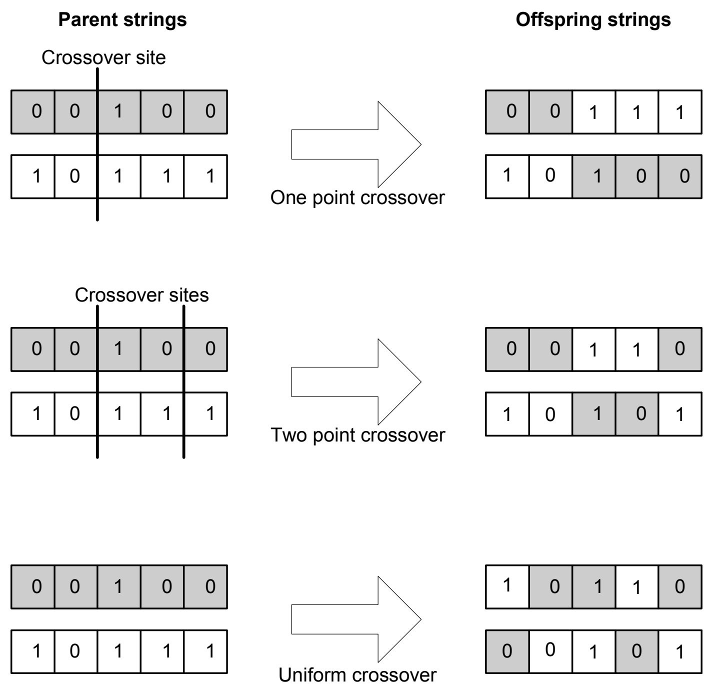


Figure 4: Illustration of one-point, two-point, and uniform crossover operators

Crossover-based GA: when the mutation operator is ignored in a GA by setting mutation probability to zero ($p_m = 0$), the GA is referred as a crossover-based GA.

Crossover Distribution Index: is any non-negative real number set by the user. This parameter affects the probability distribution of the Simulated Binary Crossover (SBX) operator. A large value of crossover distribution index gives a higher probability for creating near-parent solutions and small value allows distant solutions to be created as offspring.

Crossover Probability or Crossover Rate: ($0 \leq p_c \leq 1$) determines the amount of crossover.

Deterministic Parameter Control: is a parameter control technique that changes the parameter values during a GA's run according to some deterministic rule, which usually depends on time such as number of generations.

Depth-First Search: is an algorithm for traversing or searching a tree or graph. It is an uninformed search that progresses by expanding the root node of the search tree and going deep as quickly as possible to the leaf nodes (nodes that have no children) of the tree until a goal state is found, or until it hits a leaf node then the search backtracks and starts off on the next node.

Diversity Metric: In MOEA, a diversity metric is used to measure the uniform spread of the obtained solutions.

Dominance Relation: A solution x_a in X is said to dominate a solution x_b in X (denoted as $x_a \prec x_b$) if it is better than or equal to x_b in all objectives and at least better than x_b in one objective.

ε-Dominance Relation: A solution x_a in X is said to ε-dominate a solution x_b in X (denoted as $x_a \prec_\varepsilon x_b$) for some $\varepsilon > 0$ if $(1+\varepsilon) f_i(x_a) \le f_i(x_b)$ for all $i = 1,\ldots,k$. The ε value represents a "tolerance" allowed for the objective values. The choice of ε is problem specific and a decision maker should choose a value that suits the meaning of the objective values best.

Evolutionary Algorithm (EA): is a generic term used to indicate any population-based stochastic search algorithm that uses mechanisms inspired by biological evolution and genetic operators, such as reproduction, mutation, crossover, natural selection and survival of the fittest. Candidate solutions to the optimization problem play the role of individuals in a population, and the fitness function determines the fitness of each individual in the population. The solutions with higher fitness value will have better chances of survival and reproduction according to the evolutionary process. Evolution of the population then takes place after the repeated application of the above genetic operators. Through evolution process, better solutions are generated out of the current population of candidate solutions and this process continues until the terminated condition is satisfied such as having one or more individuals whose fitness exceeds some threshold.

Evolutionary Programming (EP): is mutation-based EA applied to discrete search spaces.

Evolutionary Strategy (ES): is an EA that uses real-coded values. Early ESs do not use any crossover-like operator. Later, crossover-like operators have been introduced into ESs. Therefore, an ES's framework is similar to that of a real-coded GA. ESs focus on optimizing continuous functions.

Genetic Algorithm (GA): is a general-purpose EA. The algorithm is well suited for optimizing combinatorial problems (though they have occasionally been applied to continuous problems) using crossover and mutation.

Genetic Drift: the condition when there is no diversity in the population to discriminate between two or more distinct individuals.

Genotype: coding or representation of individuals (chromosomes).

Goal Programming Method: In this method, decision maker have to assign goals that they wish to achieve for each objective. These values are incorporated into the problem as additional constraints. This method will then try to minimize the absolute deviations from the goals to the objectives.

Hamming Cliffs: are formed when two numerically adjacent values have bit representations that are far apart. For example, the corresponding 8-bit binary representations for two decimal numbers 127 and 128 are 01111111 and 10000000.

Hamming Distance: is the number of corresponding bits that differ in binary representations.

Hill-Climbing: is a graph search algorithm. The algorithm expands the current state in the search and evaluates its children. The best child is selected for further expansion and neither its siblings nor its parent is retained. The search stops when it reaches a state that is better than any of its children. The algorithm does not keep any history; therefore, it cannot recover from failures.

Heuristic Search: employs some rules for choosing search regions in the search space that are most likely leading to an acceptable problem solution (Luger, 2002).

Indifferent: The solution x_a is said to be indifferent to a solution x_b, if neither solution is dominating the other one.

Mating Pool: In GA, fitness selection is performed separately before genetic operations. Individuals of higher fitness in a population are placed in an area referred as the mating pool. Children are created from parents chosen only from the mating pool.

Monte Carlo: is a statistical simulation method for approximating solutions to quantitative problems. It uses a pure random search where any selected solution is fully independent of any previous solution and its outcomes. The current optimal solution and its associated decision variables are saved as a comparator. In the subsequent simulation replications, the current optimal solution may be replaced with even better one.

Multi-Objective Optimization (MOO): can be defined as the problem of finding "a vector of decision variables which satisfies constraints and optimizes a vector function whose elements represents the objective functions. These functions form a mathematical description of performance criteria, which are usually in conflict with each other. Hence, the term "optimize" means finding such a solution which could give the values of all the objective functions acceptable to the designer (Osyczka, 1985)."

Multi-Objective Evolutionary Algorithm (MOEA): is a variant of the traditional Genetic Algorithm (GA) - also known as the simple GA, designed to solve multi-objective optimization problems.

Mutation: takes a single individual and randomly changes some of its characteristics. The most commonly used are the bitwise mutations. In bitwise mutation, a bit in a binary string is changed (a 0 is converted to 1, and vise versa) with mutation probability as illustrated in Figure 5.

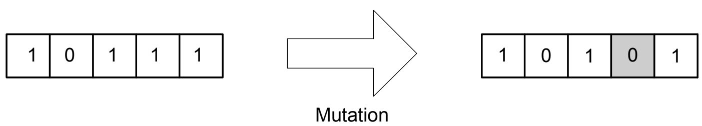


Figure 5: Illustration of bitwise mutation operator

Mutation-Based GA: when the crossover operator is ignored in a GA by setting crossover probability to zero ($p_c = 0$), the GA is referred as a mutation-based GA.

Mutation Distribution Index: is any non-negative real number set by the user. This parameter affects the probability distribution of the polynomial mutation operator (a real-coded mutation operator).

Mutation Probability or Mutation Rate: ($0 \leq p_m \leq 1$) determines the amount of mutation.

Non-Dominated: A solution is referred as non-dominated if it is not dominated by any other solutions.

Parameter Control: is a technique that monitors an EA's progress in order to adjust its parameter values during its execution.

Parameter Settings: three main parameters of a GA or MOEA are population size, crossover and mutation probabilities. The values of these three parameters need to be set properly in order to achieve desirable solutions because the performance of an EA is affected by these parameter settings.

Pareto Optimal: A vector of decision variables $\boldsymbol{x}^* \in X$ is Pareto optimal if $\boldsymbol{x}^*$ is non-dominated with respect to the set of all possible vectors in *X*. Pareto-optimal vectors are characterized by the fact that improvement in any one objective means worsening at least one other objective.

Pareto-based MOEAs: are MOEAs in which each solution is assigned a fitness value equal to its non-dominated rank in the population and ranking selection method based on the concept of Pareto optimality.

Pareto-Optimal Front: The plot of the objective functions whose non-dominated solutions are in the Pareto-optimal set is called the Pareto-optimal front.

Pareto-Optimal Set: is the set of all possible Pareto-optimal solutions.

Particle Swarm Optimization (PSO): simulates a bird flock's behavior where social sharing of information takes place and individuals can profit from the discoveries and previous experience of all other companions during the search for food. A population (swarm) of particles is assumed to fly over the search space in order to find promising regions of the landscape. For example, in the minimization case, such regions have lower functional values than other previously visited regions. In this context, each particle is treated as a point in an *n*-dimensional space, which adjusts its own flying behavior according to its flying experience as well as the flying experience of other particles. PSO, to some extent, resembles GAs. In PSO, instead of using genetic operators, each particle updates its own position based on its own search experience and other particles' experience and discoveries (Magoulas, Eldabi, & Paul, 2002).

Phenotype: the meaning of a particular chromosome as interpreted by the user (solutions in its search space).

Polynomial Mutation Operator: a real-coded mutation operator developed by Deb & Goyal (1996). This operator mutates a real-valued variable vector that simulates the way mutation work in binary-coded GAs.

Population Size: is the number of individuals or candidate solutions in the population. This parameter is a major factor in determining the quality of the solutions. Setting the population size not large enough will cause a GA to converge to sub-optimal solutions. On the other hand setting the population size above optimum will cause a GA to waste unnecessary computational resources.

Random Search: is the simplest stochastic search method. It simply evaluates a given number of randomly selected solutions.

Random Walk: is similar to random search except that the search moves iteratively from a current solution to the next randomly selected solution in the neighborhood of the current solution.

Real-Coded GA: a GA that is based on real-valued representation.

Real-Coded Representation or Real-Valued Representation: decision variables are used as it without coding. Each gene represents a decision variable of the problem. A chromosome is represented as a vector of decision variables in real-valued numbers.

Recombination: same as crossover.

Recombination Probability or Recombination Rate: same as crossover probability.

Running Performance Metric: refers to a performance metric executed during a MOEA's run on the obtained solutions instead of at the end of its run on the final obtained solutions.

SBX: is an acronym of Simulated Binary Crossover.

Schema: A schema is a similarity template describing a subset of strings with similarities at certain positions (Holland, 1975; Goldberg, 1989). A schema for binary strings can be defined over the triplet (0, 1, *). The symbol '*' represents a don't- care symbol. For example, a schema 01*** represents eight binary strings with a '0' in the first position and a '1' in the second position. A string that matches a schema's pattern is called an instance of that schema. For example, 01011 and 01000 are both instances of the schema 01***. In a schema, 0's and 1's are referred to as *defined bits or alleles.* The *fixed positions* of a schema are the positions that contain a '0' or a '1'. The number of defined bits or fixed positions in a schema is the *order* of that schema. A schema's *defining length* is the distance between the leftmost and rightmost defined bits. For example, the defining length of *10*1 is three, the defining length of 01*** is one, and the defining length of ***1* is zero. Since a schema represents a subset of strings, it can also be viewed as representing certain region in the search space. The average fitness of a schema is determined by the average fitness of instances of the schema.

Search Algorithm: broadly speaking, is an algorithm that takes a problem as input and returns a solution to the problem, usually after evaluating a number of possible solutions.

Search Space: is the set of all possible solutions to a problem.

Self-Adaptive Parameter Control: is a parameter control technique that adjusts the values of parameters attached to each individual in the population according to information fed back from the GA during its run. The parameters are co-evolved with each individual in the population and each individual in the population can have different parameter values.

Selection: In GA, selection is a process of identifying individuals of higher fitness in a population and placing them in the mating pool for reproduction.

Selection Pressure or Selection Rate: determines the number of best individuals to be placed in the mating pool after the selection operation.

Simulated Binary Crossover (SBX): is a real-coded crossover operator developed by Deb & Agrawal (1995). The SBX operator creates a new pair of offspring vectors from a pair of real-valued parent vectors. It simulates the working principle of the single-point crossover operator on binary strings.

Simulated Annealing (SA): The name comes from the analogy to the behavior of physical systems by melting a substance and lowering its temperature slowly until it reaches freezing point (physical annealing). SA avoids getting stuck in local optima as in hill climbing) by accepting all downhill moves (assume here a minimization problem) but sometimes accepting uphill moves, where the acceptance probability decreases to 0 at a certain rate (Carson & Maria, 1997; Fu, 2001).

Single Point Crossover or One Point Crossover: is the simplest form of crossover. A single crossover site is chosen at random and the components of two parents after the crossover site are swapped to produce two offspring (see Figure 4).

Standard Parameter Settings: A good set of parameter settings for simple GAs introduced by De Jong (1975) that have been adopted widely: population size of 50-100, crossover probability of 0.6, and mutation probability of 0.001. However, these standard settings have been proven problematic.

Stochastic Search: search process that involves some type of randomness.

Tabu Search: is a variation of local search that maintains a fixed-length of explored moves. This list contains the forbidden (tabu) moves that are not allowed at the present iteration in order to exclude back tracking moves (Carson & Maria, 1997).

Tournament Selection: In *n*-wise tournament selection, *n* individuals are selected at random for a tournament and the best individual is selected. The tournament selection is repeated until the mating pool is full.

Trade-Off Solution: The term "trade-off" in the MOEA context refers to the fact that a value of one objective function is traded for a value of another function or functions.

Weighted Min-Max Method: In this method, the decision maker defines the min-max optimum (the set of points that give the smallest values of the relative deviations from the individual objective function) and applying it to a MOP.

Weighting Sum: This method consists of adding all the objective functions of a MOP together using different weighting coefficients for each one. Thus, the MOP is transformed into a scalar optimization problem of the form:

$$\min \sum_{i=1}^{k} w_i f_i(\overline{x}) \tag{4}$$

where, $w_i \geq 0$ are the weighting coefficients representing the relative importance of the objectives. It is usually assumed that

$$\sum_{i=1}^{k} w_i = 1 \tag{5}$$

Summary

This chapter presented an overview of the problem, goal, and domain focused on by the dissertation. It started by providing a brief overview of a Multi-Objective Problem (MOP) and Multi-Objective Evolutionary Algorithms (MOEAs). It then described the problem that the dissertation investigates and the goal that the dissertation accomplishes. MOEAs are not easy to use because they require parameter tunings of population size, crossover probability, and mutation probability in order to achieve the desirable solutions and performance for an arbitrary complex problem. The task of tuning these parameters is not trivial due to due to the complex and nonlinear interactions among the parameters and their dependency on many aspects of the particular problem being solved such as the search space size and the shape of the fitness surface. Moreover, the proper parameter values are not fixed but varied during a run because a MOEA is dynamic and adaptive process. This dissertation aims to investigate simultaneous parameter control techniques in MOEA for all three parameters - population size, crossover, and mutation. The goal of this dissertation is to develop a MOEA with adaptable population size crossover, and mutation, parameters for automating the process of selecting appropriate parameter values in order to make the MOEA more efficient, easier to use and available to more users.

The chapter then describes why research in parameter control techniques for MOEAs is relevance and significant. Previous research indicated that the issue of controlling values of various parameters of a GA is one of the most important and promising areas of research in evolutionary computation. It has a potential of adjusting the algorithm to the problem while solving the problem and the user does not have to specify or tune the values of the parameters. Thus, it makes GAs more efficient, easier to use and available to more users. Next, this chapter presents the known barriers and issues of MOEAs including: individuals of the population converges towards different areas of the Pareto front and efficient parameters differ between the individuals; MOEAs work well for approximation of the Pareto front, but fail in the final convergence since the archive size of this class of algorithms is usually limited; the adaptive population size with multiple population approach, which is applied successfully in the simple GA, does not work well for a Pareto-based MOEA like the NSGA-II; Pareto-optimal solutions may be replaced by other inferior non-dominated solutions due to the way the algorithm preserve elitism in the NSGA-II; the elitism of NSGA-II requires a population of size $2N$, instead of size N required in most other MOEAs; lack of reliable and efficient running metrics, and lack of theoretical proofs. The chapter then lists research questions that the dissertation provides the answers followed by limitations and delimitations. Finally, this chapter provides definition of terms that are used throughout the dissertation.

Chapter 2

Review of the Literature

This chapter provides a review of research literature. It is organized as follows: First, it presents an historical overview of MOEAs. Second, it reviews research literature specific to the dissertation topic. Third, it provides a summary of known an unknown about the research topic. Finally, the chapter describes the contributions that this dissertation makes to the field MOEA.

Historical Overview

This section presents an historical overview of multi-objective evolutionary algorithms (MOEAs). First, traditional methods used for optimizing multi-objective problems and the weaknesses associated with these methods are described. Second, evolutionary algorithms with focus on genetic algorithms are presented. Third, multi-objective evolutionary algorithms, which are variants of evolutionary algorithms used for optimizing multi-objective problems, are introduced. The next three sections describe three popular approaches for MOEAs: aggregation-based approaches, population-based approaches, and Pareto-based approaches. The section then continues by describing the non-dominated sorting procedure and niche strategy with fitness sharing that most popular and successful Pareto-based MOEAs are based on. Finally, the section provides a brief review of the current state-of-the-art MOEAs.

Traditional Methods

Figure 6 below illustrates a schematic of an ideal multi-objective optimization procedure where there is no priori preference defined among the objectives (Deb, 2001). In Step 1 (vertically downwards), a MOP is fed into an ideal multi-objective optimization algorithm and a set of wide spread Pareto-optimal solutions is found. Thereafter, in Step 2 (horizontally, towards the right), a decision maker or a decision support system uses higher-level information to choose one of the trade-off solutions.

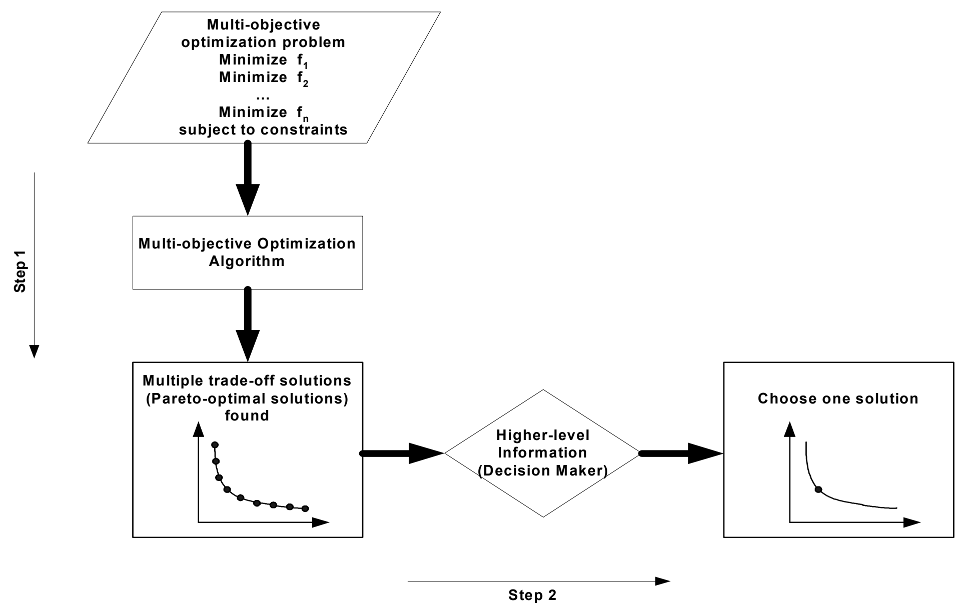


Figure 6: Schematic of an ideal multi-objective optimization procedure

Looking back at Figure 2 (in Chapter 1 - Introduction), one can observe that each Pareto-optimal solution corresponds to a specific order of importance of the objectives. For example, solution A assigns more importance to premium than to loss. On the other hand, solution B assigns more importance to loss than to premium. If such a relative preference factor among the objectives is known for a specific problem, it is not necessary to follow the step 2 in Figure 6 for solving a MOP. In this case, the straightforward approach to handle multi-objectives with any multi-objective optimization algorithm is to combine all objectives of a problem into a single-objective with a relative preference factor among the objectives then use the algorithm to solve the problem. There are several approaches of combining multiple objectives into a single-objective and these approaches are known as aggregating functions. The aggregating functions are traditional approaches, which have been used successfully in the field of operation research to solve MOPs when the behavior of the objective functions is known (Coello, 2000; Deb, 2001). The most popular aggregating approaches include weighted sum, ϵ-constraint, goal programming, and goal attainment (Coello, 2000). Typically, a relative preference vector of weighted objective values is used to convert the MOP into a composite single-objective problem as the weighted sum of the objectives, where a weight for an objective is proportional to the preference factor assigned to that particular objective. When such a composite objective function is optimized, it is possible to obtain one particular trade-off solution. A change in the preference vector will result (hopefully) in a different trade-off solution. One obvious weakness of these aggregating approaches is that it may be difficult to supply a relative preference vector that properly scales the objectives when little is known about the problem. Unless an accurate preference vector is available, the trade-off solution obtained by this approach is highly subjective to a particular user. However, the most serious weakness of these approaches is that they cannot find a good trade-off solution to all problems, especially when the Pareto front is concave, regardless of the preference vectors used (Coello, 2001; Deb, 2001). This dissertation concerns with the ideal multi-objective optimization approach as illustrated in Figure 6, which is less subjective, more methodical, and a user does not need to know about any priori preference information.

Using exact search methods, such as hill-climbing, depth-first, breadth-first, best-first (Section “Definition of Terms” in Chapter 1 provides definitions of these search methods) as multi-objective optimization algorithms for solving MOPs, can be computationally expensive and is often infeasible in finding the Pareto-optimal set because the complexity of the multi-objective problem being optimized prevents these methods from being applicable. For this reason, a number of stochastic search methods, which use some type of randomness in the search process, such as random search, tabu search, simulated annealing, Monte Carlo, particle swarm optimization, evolutionary algorithms, and others have been developed and used for solving MOPs (Section “Definition of Terms” in Chapter 1 provides definitions of these stochastic search methods). These stochastic methods usually do not guarantee to find the true Pareto-optimal set but try to find a good set of solutions, which are hopefully close to the true Pareto-optimal set. Of these stochastic search methods, random search, tabu search, simulated annealing, Monte Carlo are single-solution based methods (i.e. they operate on a single solution and can find only one solution at a time). Whereas, particle swarm optimization and evolutionary algorithms operate on a population of candidate solutions, which are more suitable for finding a set of good trade-off solutions in the ideal multi-objective optimization approach described in Figure 6 above. Which method is the best depends upon the problem being solved. In general, there is no universal best algorithm, which can achieve performance advantage for all problems (Corne & Knowles, 2003). This dissertation chooses to focus its discussion on using evolutionary algorithms to solve MOPs, namely Multi-Objective Evolutionary Algorithms (MOEAs). The EAs are described next.

Evolutionary Algorithms

Evolutionary Algorithm (EA) is a generic term used to indicate any population-based stochastic search algorithm that uses mechanisms inspired by biological evolution and genetic operators such as reproduction, mutation, crossover, natural selection and survival of the fittest. Candidate solutions to the optimization problem play the role of individuals in a population, and the fitness function determines the fitness of each individual in the population. The solutions with higher fitness value will have better chances to survive and reproduce better solutions according to the evolutionary process. There are three main types of Evolutionary Algorithms (EAs): Genetic Algorithm (GA) (Holland, 1975), Evolutionary Strategy (ES) (Rechenberg, 1965), and Evolutionary Programming (EP) (Fogel, Owens, & Walsh, 1966). These algorithms were developed independently but their general framework is essentially the same. They evolve a population of candidate solutions to a given problem in order to find optimal solutions, using operators inspired by natural genetic variation (creating new individuals by means of crossover and mutation), natural selection and survival of the fittest (competition for reproduction and resources among individuals) in the biological world. However, each type has some variants due to different origins and their targets towards specific domains. GAs use crossover and mutation, and they are well suited for optimizing combinatorial problems (though they have occasionally been applied to continuous problems). ESs use real parameter values and early ESs do not use any crossover-like operator. Later, crossover-like operators have been introduced into ESs. Therefore, an ES's framework is similar to that of a real-parameter GA. ESs focus on optimizing continuous functions. EPs are mutation-based evolutionary algorithms applied to discrete search spaces. Since most existing MOEAs are based on the GA due to its flexibility in solving complex optimization problems, this dissertation narrows its discussion on the GA. Therefore, the GA is described in more detail in the following.

Genetic Algorithms (GAs)

Genetic Algorithms (GAs) are general-purpose stochastic and heuristic search algorithms that mimic the evolutionary process in order to find the fittest solutions. The algorithms were invented by John Holland in the 1960s and presented in his pioneering book (Holland, 1975). Over the years, the Holland's original GA (also known as simple GA or traditional GA) has evolved into many forms such as distributed GA, parallel GA, multi-modal GA, and others. However, the general framework remains the same as in the simple GA. Pseudo-code for the simple GA is presented in Table 1.

Table 1: Pseudo-code for the simple GA

*Generate a **population** of random solutions*
Repeat
    ***Evaluate** of the fitness of each solution in the population*
    ***Select** solutions with high fitness for reproduction*
    *Apply genetic operators **crossover** and **mutation** to generate new solutions*
Until the terminate conditions are satisfied

A simple GA attempts to find a good solution to some problem (e.g., finding the minimum of a function) by generating a population of candidate solutions to the problem and then manipulating these solutions using genetic operators. The initial population of candidate solutions can be generated randomly or by using prior knowledge of possibly good solutions. Without any knowledge of the problem domain, the GA begins to process population of solutions. The function to be optimized can be used as the fitness function to evaluate each solution. Each individual in the population is evaluated and ranked based on its fitness. The solutions with higher fitness value will have better chances of survival and reproduction according to the evolutionary process. Hence, these individuals will be selected to produce the subsequence generation of candidate solutions. There exist a number of different selection operators such as proportionate selection, ranking selection, and tournament selection. The tournament selection operator is commonly used due to its simplicity and controlled takeover property (Goldberg & Deb, 1991). In *n*-wise tournament selection, *n* individuals are selected at random for a tournament and the best individual is selected into the mating pool. The tournament selection is repeated until the mating pool is full. Two candidate solutions with high fitness are selected randomly from the mating pool. Crossover operator is then used to swap genes of the selected parent solutions in order to produce new pair of offspring that are likely to be better solutions. Mutation operator is used to change some characteristic of an individual. Mutation operator is often used to maintain diversity in the population in order to prevent the GA from being trapped in local sub-optima. Through these genetic operations, better solutions are generated out of the current population of candidate solutions and this process continues until the terminated condition is satisfied such as having one or more individuals whose fitness exceeds some threshold.

GAs have received growing interest due to their simplicity as algorithms and their ability to discover good solutions quickly for complex searching and optimization problems involving features such as discontinuities, multi-modality, disjoint feasible spaces, and noisy function evaluations. The algorithms work with a population of potential solutions so they can offer a number of possible solutions. Their potential applications are numerous. They have been used in a wide range of applications including but not limited to: optimization of functions with linear and nonlinear constraints, classification, machine learning, parallel semantic networks, simulation of gas pipeline systems, problems of scheduling, web search, software testing, financial forecasting, ecology, and social systems.

Parameter Settings in Genetic Algorithms

GAs are not easy to use because they require parameter tunings in order to achieve the desirable solutions and performance. Three common GA parameters that a GA user often has to tune are population size, crossover probability, and mutation probability. The population size parameter is a major factor in determining the quality of the solutions. Setting the population size not large enough will cause the GA to converge to sub-optimal solutions. On the other hand setting the population size too large will cause the GA to waste unnecessary computational resources. The crossover probability parameter ($0 \leq p_c \leq 1$) determines the amount of gene swapping between the parent solutions. Cross operator is important because it ensures good mixing of candidate solutions. The higher crossover probability, the more promising solutions are mixed. This also increases the disruption of good solutions. The mutation probability parameter ($0 \leq p_m \leq 1$) determines the amount of mutation on a solution. Mutation operator is important because it enables diversity in the population. With a high mutation probability, mutation represents a random search, similar to an intelligent hill-climbing strategy, in the neighborhood of a particular solution, but it may also destroy already found good solutions. The task of tuning these GA parameters has been proven to be far from trivial due to the complex interactions among the parameters and their proper settings cannot be independently determined. Many researchers have been trying to understand the interdependencies of GA parameters. One of the first empirical studies to understand the complex interactions and interdependencies of GA parameters was investigated by De Jong (1975). Based on his studies, De Jong introduced a good set of parameter settings that have been adopted widely and sometimes referred to as “standard” settings: population size of 50 to 100, crossover probability of 0.6, and mutation probability of 0.001. However, these “standard” settings have been proven problematic by later studies, which suggest that the optimal settings of GAs’ parameters are critically dependent on the nature of the function being evaluated (Goldberg, 1985; Hart & Belew, 1991; Deb, 1999a) and the encoding of decision variables (Battle & Vose, 1990; Radcliffe, 1991; Karguta, Deb, & Goldberg, 1992; Tate & Smith, 1993). Hence, the choice of GA parameter settings itself can be a complex nonlinear optimization problem (Srinivas & Patnaik, 1994). As a result, it takes trial and error experiments to obtain the proper GA parameter settings for an arbitrary real-world problem. Moreover, proper parameter values are not fixed but varied during a run because a GA is dynamic and adaptive process (Fogarty, 1989; Davis, 1991; Hesser & Männer, 1991; Bäck, 1992; Mühlenbein, 1992). Therefore, the use of fixed parameter settings may lead to slow convergence and sub-optimal obtained solutions, especially when large search spaces are to be explored in solving complex optimization problems.

Multi-Objective Evolutionary Algorithms (MOEAs)

A Multi-Objective Evolutionary Algorithm (MOEA) is a modified version of the simple GA, designed to solve MOPs. The potential of using EAs to solve MOPs was originally hinted by Rosenberg in his dissertation (Ronsenberg, 1967). He suggested using a GA for finding the chemistry of a population of single-celled organisms with multiple properties or objectives. Since a GA works with a population of candidate solutions, the dominance criteria can be used to gear the search process toward the

Pareto-optimal front and multiple Pareto-optimal solutions can be obtained in a single run. This is the main reason that makes GAs ideally suitable for multi-objective optimization.

In contrast to single-objective optimization, where objective function and fitness function are often the same, in multi-objective optimization, both fitness assignment and selection must support several objectives. Therefore, MOEAs varies from the simple GA only in the way fitness assignment and selection works. Several different versions of MOEAs have been introduced with different fitness assignment and selection strategies. Based on their fitness assignment and selection strategies, MOEAs can be categorized as aggregation-based approaches, population-based approaches, and Pareto-based approaches (Coello, 2000; Zitzler, Laumanns, & Bleuler, 2004). These approaches are described briefly in the following.

Aggregation-based Approaches for MOEAs

Since the simple GA relies on a scalar fitness function to guide the search, the most intuitive approach for using a GA to solve a MOP is to combine all objectives of a problem into a single-objective problem using one of the traditional aggregating-functions methods described previously. Then the GA is used to solve the problem. An example of this approach is a linear sum of weights (known as weighting sum), which consists of adding all the objective functions together using different weighting coefficients for each one. Thus, the multi-objective optimization problem is transformed into a scalar optimization problem of the form:

$$\min \sum_{i=1}^{k} w_i f_i(\overline{x}) \tag{6}$$

where, $w_i \geq 0$ are the weighting coefficients representing the relative importance of the objectives. It is usually assumed that $\sum_{i=1}^{k} w_i = 1$.

Aggregating functions can be linear (as in the weighting sum example above) or non-linear such as aggregating functions adopted by game theory (Rao, 1987), goal programming (Wienke, Lucasius, & Kateman, 1992; Deb, 1999c), goal attainment (Wilson & Macleod, 1993; Zebulum, Pacheco, & Vellasco, 1998), and min-max algorithm (Hajela & Lin, 1992; Coello & Christiansen, 1998). Aggregation-based approaches do not require any changes to the basic mechanism of a simple GA. Therefore, they are efficient, simple, and easy to implement. They can be used to solve simple multi-objective optimization problems with few objective functions and convex search spaces. However, the aggregation-based MOEA approaches suffer from the following difficulties (Deb, 2001):

- A Pareto-optimal solution is specific to the preference parameters used in converting a MOP into a single-objective optimization problem. In order to find a different Pareto-optimal solution, the preference parameters must be changed and the new single-objective optimization problem has to be solved again. Thus, in order to find n different Pareto-optimal solutions, at least n different single-objective optimization problems need to be formed and solved.
- They are sensitive towards the preference vector of weighted objective values.
- They require the user to have some knowledge about the problem being solved in order to generate the preference parameters.

- Some aggregating-functions methods are sensitive to the shape of the Pareto-optimal front (e.g. the weighted sum method cannot find a good trade-off solution to all problems when the Pareto front is concave).

Population-based Approaches for MOEAs

This class of MOEAs switches between the objectives during the selection phase. Each time an individual is selected for reproduction, potentially a different objective will decide which member of the population will be copied into the mating pool. The Vector Evaluated Genetic Algorithm (VEGA) is one of the examples of these approaches. The VEGA, which is also the first actual implementation of what it is now called a multi-objective evolutionary algorithm (MOEA), was introduced by David Schaffer in 1984 (Schaffer, 1985), mainly intended for solving machine learning problems. It is a simple GA with a modified selection strategy. A loop is added around the traditional selection procedure so that the selection method is repeated for each objective to fill up a portion of the mating pool. With this proportional selection, at each generation a number of sub-population is generated. For a problem with k objectives and a population size N, k sub-populations of size N/k each would be generated. These sub-populations would be shuffled together to obtain a new population of size N. The GA then applies the crossover and mutation operators on the new population in the usual way. Since only the selection mechanism of the GA needs to be modified, the VEGA is easy to implement and quite efficient. However, the solutions generated by the VEGA are often locally non-dominated because the non-dominance is limited to the current population at each generation. VEGA also tends to bias toward some particular objectives. These problems occur because the algorithm selects solutions with high fitness in one objective, without looking at the others (Coello, Pulido, & Montes, 2005). As a result, the VEGA is able to find a Pareto-optimal set but fails to obtain a good spread of solutions.

Regardless of the limitations of the population-based approaches, the simplicity of these approaches has attracted several researchers (Coello, 2005). Researchers have introduced variations of VEGA or other similar population-based approaches (see (Venugopal & Narendran, 1992; Sridhar & Rajendran, 1996; Norris & Crossley, 1998; Rogers, 2000)).

Pareto-based Approaches for MOEAs

The idea of assigning an individual's fitness based on Pareto dominance in order to overcome the problems associated with VEGA was initially proposed by David Goldberg in his 10-line sketch of a non-dominated sorting procedure (Goldberg, 1989). In the non-dominated sorting procedure, a ranking selection method based on the concept of Pareto optimality is used to assign non-dominated solutions in a population and a niche strategy with fitness sharing is used to maintain good spread of solutions among a non-dominated ranking class. The non-dominated sorting procedure and the niche strategy are described in the following.

Non-dominated Sorting Procedure and Niche Strategy with Fitness Sharing

Non-Dominated Sorting Procedure: This procedure assigns each solution a fitness value equal to its non-dominated rank in the population instead of its absolute fitness value. The procedure starts to find the set of solutions that is Pareto-optimal in the current population. These solutions are assumed to constitute the first Pareto-optimal front; therefore, the highest rank is assigned to the solutions (the same rank is assigned to all of them). In order to maintain diversity in the solutions, niche strategy (described in the next paragraph) is applied. Afterward, the solutions of the first front are temporarily discounted. The Pareto-optimal solutions from the remaining population are assigned the next highest rank. The process continues until all the solutions in the population are ranked. After the ranking process, a ranking selection scheme is applied. Solutions selected for reproduction are based on their non-dominated rank rather on their absolute fitness values. The non-dominated sorting procedure have been criticized for $O(mN^3)$ computational complexity (where m is the number of objectives and N is the population size), which makes the procedure inefficient for large population sizes (Deb, Pratap et al., 2002; Coello, 2003). This large complexity occurs because of the complexity involved in the non-dominated sorting in every generation. In order to sort a population of size N according to the rank of non-domination, each solution must be compared with every other solution in the population to find if it is dominated. This requires $O(mN)$ comparison for each solution. When this procedure continues to find the set of solutions in the first Pareto-optimal front for all solutions in the current population, the total complexity is $O(mN^2)$. Then, the solutions of the first front are temporarily discounted and the process is repeated to find subsequent fronts. In the worst case, where there exists only one solution in each front, the complexity of the non-dominated sorting procedure is $O(mN^3)$.

Niche Strategy with Fitness Sharing: The GA tends to converge to a single solution when used with a finite population due to stochastic errors associated with genetic operators (Deb & Goldberg, 1989). This behavior is known as genetic drift occurred when the algorithm cannot discriminate between two or more distinct solutions because there is no diversity in the population. The behavior is acceptable if the goal is to find a single good approximation of the global optimal solution. However, in a multi-objective optimization problem, the goal is generally to find several trade-off solutions. Therefore, niche strategies are introduced to maintain wide spread of solutions. One of the niche strategies which has proven effective is the fitness sharing proposed by Goldberg (1989). Fitness sharing reduces the fitness of each solution by the presence of other solutions in the same neighborhood (a niche), so that the population is balanced among multiple niches. The fitness value reduction of near individuals can be calculated using the following equations (Dias & Vasconcelos, 2002):

$$Sh(d_{ij}) = \begin{cases} 1-\left(\dfrac{d_{ij}}{\sigma_{share}}\right)^2, & \text{if } d_{ij} < \sigma_{share} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$df_i^{'} = df_i \left[\sum_{j=1}^{N} Sh(d_{ij}) \right]^{-1} \tag{8}$$

In the above equation, the parameter d_{ij} is the Euclidean distance between two solutions i and j; σ_{share} is the maximum distance allowed between any two solutions to become members of a same niche; df_i is the dummy fitness value assigned to the solution i in the current front and $df_i^{'}$ is its corresponding shared fitness value; and N is the population size. The fitness sharing technique introduces an additional parameter σ_{share}. This parameter is usually set by the user and the performance of the sharing function method in maintaining good spread of solutions largely depends on the chosen σ_{share} value (Deb, Pratap et al., 2002).

Many researchers have developed different versions of MOEAs based on the concept of Pareto optimality such as Multi-Objective GA (MOGA) (Fonseca & Fleming, 1993), Strength Pareto EAs (SPEAs) (Zitzler & Thiele, 1998; Zitzler et al., 2002), and Non-dominated Sorting GAs (NSGAs) (N. Srinivas & Kalyanmoy Deb, 1994; Deb, Pratap et al., 2002). MOGA uses the dominance rank, i.e., the number of individuals by which an individual is dominated, to determine the fitness values. SPEA and SPEA2 calculate fitness values based on both dominance rank and dominance count, i.e. the number of individuals dominated by a certain individual. NSGA and NSGA-II use the dominance depth to assign the fitness values (i.e. the population is divided into several fronts and the depth reflects to which front an individual belongs to). Regardless of the fitness strategy used, a fitness value is related to the whole population in contrast to other approaches, which assign an individual's fitness value independently of other individuals (aggregation-based approaches) or calculate an individual's fitness value is limited to the current population at each generation (population-based approaches). The main problem with Pareto-based approaches is that there exists no efficient and reliable metric to evaluate the convergence of obtained solutions towards the true Pareto-optimal set. In addition, there lack theoretical proof for convergence to the true Pareto-optimal set and diversity of the obtained non-dominated set. Deb (1999b) has shown that solutions obtained by Pareto-based MOEAs are not always the true Pareto-optimal solutions but non-dominated solutions. However, in most complex test problems and real-world applications tried so far most of the Pareto-based MOEAs have found the true Pareto-optimal set or near the true Pareto-optimal set (Deb, 2003). Although Pareto-based MOEAs are different from each other, they have the same general framework. The Pareto dominance ranking and the niche strategy processes are executed before the ranking selection operation as explained previously. Thereafter, selection, crossover, and mutation operators are carried out as usual. Pseudo-code for a Pareto-based MOEA is presented in Table 2 below.

Table 2: Pseudo-code for a Pareto-based MOEA

```
Generate a population of random solutions
Repeat
    Evaluate of the fitness of each solution in the population
    Front = 1
    Repeat
        Find non-dominated solutions in the current population and
        Rank Pareto-optimal front to the found non-dominated solutions
        Apply niche strategy to maintain diversity in the solutions
        Remove non-dominated solutions in the current front from further contention
        Front = Front + 1
    Until all the solutions in the population are ranked
    Select solutions based on non-dominated rank for reproduction
    Apply genetic operators crossover and mutation to generate new solutions
Until the terminate conditions are satisfied
```

Current State-Of-The-Art MOEAs

Until to date, most of the successful MOEAs are Pareto-based approaches derived from the non-dominated sorting procedure (Deb, 2003; Coello, 2005). There have been two generations of MOEAs as described in the following:

First Generation MOEAs: The first generation of MOEAs is typically characterized by the use of selection mechanism based on Pareto ranking and fitness sharing to maintain diversity. The most representative MOEAs from the first generation include Non-dominated Sorting Genetic Algorithm (NSGA) (N. Srinivas & Kalyanmoy Deb, 1994), Niched-Pareto Genetic Algorithm (NPGA) (Horn, Nafpliotis, & Goldberg, 1994), and Multi-Objective Genetic Algorithm (MOGA) (Fonseca & Fleming, 1993). The first generation MOEAs have the following main issues (Coello et al., 2005):

- Are there another ways to maintain diversity in the population without using fitness sharing, which requires $O(N^2)$ where N refers to the population size?

- Is there another way to design a more efficient MOEA that reduces the $O(mN^3)$ process required to perform non-dominated ranking (m is the number of objectives and N is the population size)?
- Are there appropriate test problems and metrics to evaluate quantitatively an MOEA? During the first generation, most comparisons were practically done visually by plotting the Pareto-optimal fronts produced by different MOEAs.
- Will there be some theoretical foundations for MOEAs?

Second Generation MOEAs: The second generation of MOEAs, which is still progressing nowadays, can be characterized by an emphasis on efficiency and by the use of elitism. In the context of multi-objective optimization, elitism usually (although not necessarily) refers to the use of a secondary population (also known as external population) to retain non-dominated solutions. Elitism can also be implemented by using (μ+λ)-selection in which parent solutions compete with their children and those solutions that are non-dominated and possibly meet some additional criterion such as having better diversity of solutions are selected for the next generation. During the second generation, some important theoretical studies have also been carried out, mainly related to convergence to the true Pareto-optimal set (Rudolph, 1998; Veldhuizen & Lamont, 1998; Hanne, 2000b, 2000a; Rudolph & Agapie, 2000; Rudolph, 2001). In addition, metrics and standard test sets have been introduced to evaluate new MOEAs (Veldhuizen, 1999; Zitzler, Deb, & Thiele, 2000). The most representative MOEAs from the second generation include Strength Pareto Evolution Algorithm (SPEA) (Zitzler & Thiele, 1999), Strength Pareto Evolution Algorithm 2 (SPEA2) (Zitzler et al., 2002), Pareto Archived Evolution Strategy (PAES) (Knowles & Corne, 2000), Non-dominated Sorting Genetic

Algorithm II (NSGA-II) (Deb, Pratap et al., 2002), Niched Pareto Genetic Algorithm 2 (NPGA 2) (Horn et al., 1994), Pareto Envelope-based Selection Algorithm (PESA) (Corne, Knowles, & Oates, 2000), and Micro Genetic Algorithm (Coello & Pulido, 2001). The second generation MOEAs have of the following main issues (Coello et al., 2005):

- How does the secondary population interact with the main population?
- How do MOEAs handle the case when the secondary population is full?
- Should additional criteria be imposed on retaining solutions in the secondary population instead of just using Pareto dominance?
- Are the performance metrics reliable? Are the test problems reliable?
- Can the current state-of-the-art MOEAs tackle problems with more than two objectives functions efficiently? Will Pareto dominance fail when dealing with too many objectives? If so, then what is the maximum limit which Pareto ranking can be used to select non-dominated solutions reliably?
- What are the most relevant theoretical aspects of MOEA that are worth exploring?

MOEAs have been used increasingly in a wide range of real-world multi-objective optimization applications including but not limited to: telecommunication network design (Flores et al., 2003; Maple et al., 2004), software quality enhancement (Khoshgoftar, 2004), risk-based corrective action design (Gopalakrishnan et al., 2001), optimization of corrugated bulkhead forms (Yang & Hwang, 2002), digital filter design (Schnier et al., 2001). Today, the MOEA repository (http://www.lania.mx/~ccoello/EMOO) contains over 2178 papers, from which a vast majority are applications (Coello, 2005).

Research Literature Specific to the Topic

This section reviews research literature specific to the topic of this dissertation. It is organized as follows: First, it presents research efforts in parameter control methods for simple GAs. Second, it describes the Non-dominated Sorting Genetic Algorithm II (NSGA-II), which the ANSGA-II is built upon. Third, this section presents some recent research efforts in parameter control methods for MOEAs. Finally, the section reviews research literature on running performance metrics.

Parameter Control Techniques in Simple GAs

Several parameter control methods have been proposed and applied successfully for single objective optimization problems using simple GAs. Both theoretical and empirical studies have suggested that the good mutation probability, which enables diversity in the population without destroying already found good solutions, varies with evolutionary time according to the state of the search and the nature of the search problem (Fogarty, 1989; Hesser & Männer, 1991; Bäck, 1992; Mühlenbein, 1992). Davis (1991) applied a time-varying schedule of parameter settings and found that performance was improved. Spears (1991) applied self-adaptation for selecting optimal crossover operator (uniform crossover or two-point crossover) and showed that this adaptive crossover operator out-performs non-adaptive crossover operator, especially with large population sizes. Smith & Fogarty (1996) used self-adaptation for mutation rate and showed that the self-adaptive mutation significantly improves the GA's performance as well as making it possible to remove the mutation parameter from the set of decisions faced by the user.

Harik and Lobo (1999) proposed a parameter-less GA to make simple GAs easier to use and available to more users. The main contribution of this work is the adaptive population size technique, which is used in the ANSGA-II. Harik and Lobo focused their study on the adaptive population size technique; therefore, they chose to ignore the mutation (p_m = 0) and set the crossover probability to a fixed value (p_c = 0.5). The parameter-less GA selects the right population size by establishing a race among multiple populations of various sizes. It allows the smaller populations more generations to run. The coordination of the array of populations is implemented with a counter base 4 as illustrated in Table 3 (Harik & Lobo, 1999).

Table 3: The coordination of the array of populations in the parameter-less GA

Counter base 4	Action
0	Run 1 generation of population 1
1	Run 1 generation of population 1
2	Run 1 generation of population 1
3	Run 1 generation of population 1
10	Run 1 generation of population 2
11	Run 1 generation of population 1
12	Run 1 generation of population 1
13	Run 1 generation of population 1
20	Run 1 generation of population 2
21	Run 1 generation of population 1
22	Run 1 generation of population 1
23	Run 1 generation of population 1
30	Run 1 generation of population 2
31	Run 1 generation of population 1
32	Run 1 generation of population 1
33	Run 1 generation of population 1
100	Run 1 generation of population 3
101	Run 1 generation of population 1
…	…

Overall, the *n*th population is allowed to run 4 times more generations than the *n*+1th population. Each time a population yields its execution if the next population has yet not been created the algorithm creates a new population twice as large as the previous one. Since the smaller populations have more time to run, they expect to converge faster than the larger one and the algorithm would terminate with a proper population size. On the other hand, if at any point in time, a larger population has an average fitness better than that of a smaller population then the algorithm would reset the counter and stop running the smaller population because it is very unlikely that the smaller population would produce better solutions than the larger one. This process continues and the algorithm would eventually terminate with a proper population size. The parameter-less GA has been tested on three test problems (the one-max problem, the noisy one-max problem, and the bounded deceptive problem) (Harik & Lobo, 1999) and applied to a real-world problem successfully (Lobo & Goldberg, 2001). A worst-case analysis of the parameter-less GA has also been studied and it demonstrates that the parameter-less GA does not increase significantly the computational requirements of the GA with optimal parameter settings (Pelikan & Lobo, 1999). The experimental studies suggested that the parameter-less GA indeed offers an efficient method to eliminate the population size parameter. The parameter-less GA is a bit slower than a GA that starts with proper parameter settings. However, it takes trial and error experiments to obtain the proper parameter settings for an arbitrary real-world problem.

Bäck, Eiben, & van der Vaart (2000) implemented simple GAs that have one or all three parameters (p_m, p_c, and N) adjusted during the run. This was the first empirical study that investigates simultaneous parameter control techniques in simple GAs for all three parameters - mutation, crossover, and population size. The self-adaptive mutation probability is encoded as extra bits at the end of every individual and initialized to random values between 0.001 and 0.25. Mutation is performed in two steps. First, only the encoded bits for the mutation probability are mutated and immediately decoded to establish the new mutation probability. This new mutation probability is then applied to the main bits (those encoding a solution) of the individual. The self-adaptive crossover probability is also encoded as extra bits at the end of every individual and initialized to random values between 0.0 and 1.0. When an individual in the population is selected for reproduction by the tournament selection, a random number *r* between 0.0 and 1.0 is compared with the individual's p_c. If *r* is less than p_c, the individual is ready to mate. If both selected individuals are ready to mate, two children are created by uniform crossover, mutated and inserted into the population. If *r* is greater than p_c, the individual will only be subject to mutation to create one offspring, which is inserted in the population immediately. If one individual is willing to mate and the other one is not, the willing parent is on hold and the next selection round only selects one other parent. The adaptive population size is implemented as described in the following. Every new individual is allocated a remaining lifetime (RLT) according to its fitness by the following bi-linear formula:

$$RLT(i) = \begin{cases} MinLT + \eta \dfrac{WorstFit - fitness(i)}{WorstFit - AvgFit} & \text{if fitness(i)} \leq \text{AvgFit} \\ \dfrac{1}{2}(MinLT + MaxLT) + \eta \dfrac{AvgFit - fitness(i)}{AvgFit - BestFit} & \text{if fitness(i)} < \text{AvgFit} \end{cases} \quad (9)$$

$$\eta = \frac{1}{2}(MaxLT - MinLT)$$

In the above formula, *MinLT* and *MaxLT* stand for the allowable minimum and maximum lifetime of an individual; *fitness(i)* stands for the fitness of individual *i*; *AvgFit* stands for average fitness; *BestFit* stands for best fit; and *WorstFit* stands for worst fitness. The values for *MinLT* and *MaxLT* are set to 1 and 11 in this study because initial runs with different values indicated that these values deliver good performance. Thus, instead of selecting a population size *N*, the user has to select a maximum lifetime *MaxLT*. Each generation, the RLT of all the individuals in the population is decremented by one except for the fittest individual, whose RLT is unchanged. If the RLT of an individual reaches zero it is removed from the population. The initial population consists of 60 individuals. It is likely that eventually every individual will die of old age.

Five variants of the algorithm are tested using a carefully designed test suite of five functions: the original GA, the GA with self-adaptive mutation probability only (SAMGA), the GA with self-adaptive crossover probability only (SAXGA), the GA with adaptive population size only (APGA), and the GA with self-adaptive mutation probability, self-adaptive crossover probability, adaptive population size (SAMXPGA). Each test function is run 30 times for every one of the five variant GAs. The best fitness and the average fitness of the population are monitored until the algorithm terminates. The speed of optimization is measured by the average number of evaluation on success (i.e. how many evaluation does it take on average for the successful runs to find the optimal solution). The success rate shows how many of the runs are successful in finding the solutions. The results show that the performance of the self-adaptive parameters p_m and p_c alone is disappointing (in SAMGA and SAXGA). The most likely reason is that the algorithms take time away from finding the optimal solution to search for good parameter values. The adaptive population size alone in the APGA, on the other hand, improves the performance of the algorithm. The overall winner is the SAMXPGA with all three adaptable parameters. The authors emphasize that using adaptive population sizes prove to be the key feature to improve the algorithm and more studies on control mechanism for variable population sizes should be done.

Non-dominated Sorting Genetic Algorithm II (NSGA-II)

The NSGA (N. Srinivas & Kalyanmoy Deb, 1994), which is based on the non-dominated sorting procedure and niche strategy with fitness sharing suggested by Goldberg (1989) as described in the section "Historical Overview" above, has the following weaknesses:

- $O(mN^3)$ computational complexity of non-dominated sorting procedure (where m is the number of objectives and N is the population size);
- An additional sharing parameter (σ_{share}) that the user must set to ensure getting good spread of solutions – a parameter-less diversity preservation approach is more desirable;
- Non-elitism approach: Elitism preserves good solutions at each generation. It can improve the performance of GAs significantly and prevent the loss of good solutions found so far during the evolution process (if good solutions are not selected for reproduction or if they are destroyed by crossover or mutation).

Deb et al. (2002) proposed a fast and elitist MOEA, named NSGA-II (Non-dominated Sorting Genetic Algorithm II) to remove the above three weaknesses. It is one of the best-known MOEAs and it has been extensively used in many studies (Zitzler et al., 2002; Büche et al., 2003; Devireddy & Reed, 2004). The algorithm has been recognized to perform as well or better than other MOEAs with the same goal of finding a diverse Pareto-optimal solution set such as the Pareto-Archived Evolution Strategy (PAES) (Knowles & Corne, 1999) and Strength Pareto Evolutionary Algorithm 2 (SPEA2) (Zitzler et al., 2002). For these reasons, the NSGA-II is selected in this study and the ANSGA-II is built upon it. The major features of NSGA-II, which include low computational complexity, parameter-less diversity preservation, elitism, and real-valued representation, are reviewed in detail in the following.

Low Computational Complexity: The NSGA-II requires at most $O(mN^2)$ computational complexity, which is lower compared to $O(mN^3)$ of NSGA. The procedure for finding non-dominated front used in NSGA-II is similar to the non-dominated sorting procedure suggested by Goldberg (1989) except that a better bookkeeping strategy is used to make it more efficient. In this bookkeeping strategy, every solution from the population is compared with a partially filled population for domination instead of with every other solution in the population as in the NSGA. Initially, the first solution from the population is kept in a set P'. Thereafter, each solution p (the second solution onwards) is compared with all solutions in P' one by one. If the solution p dominates any solution q in P' then solution q is removed from P'. Otherwise, if solution p is dominated by any solution q in P', the solution p is ignored. If solution p is not dominated by any solution in P' then it is saved in P'. Therefore the set P' grows with non-dominated solutions. When all solutions of the population is checked, the solutions in P' constitute the non-dominated set. To find the other fronts, the non-dominated solutions in P' will be discounted from P and the above procedure is repeated until all solutions in P are ranked. Therefore, the domination checks requires a maximum of $O(N^2)$ because the second solution is compared with only one solution of $P^{'}$, the third solution with at most two solutions of $P^{'}$, and so on. Since each domination check requires m function value comparisons, the maximum complexity of this approach to find the first Pareto-optimal front is $O(mN^2)$.

Parameter-less Diversity Preservation: To maintain diversity among solutions, the NSGA-II replaces the fitness sharing approach in the NSGA with a crowded comparison approach, which does not require any user-defined parameter. As a result, the sharing parameter σ_{share} used in the NSGA is eliminated. In the crowded comparison approach, every solution i in the population has two attributes: a non-domination rank (i_{rank}) and a crowding distance ($i_{distance}$). The crowding distance $i_{distance}$ of a solution i is a measure of the perimeter of the largest cuboid enclosing the solutions i, without including any other solution in the population, formed by using the nearest neighbor solutions as the vertices. Figure 7 illustrates the crowding distance calculation for the solution i in its non-dominated front, which is the average side-length of the cuboid enclosing the solutions i (shown with a dash box). The crowded tournament selection operator, which is used to guide the search towards a spread-out Pareto-optimal front, is defined as follows (Deb, 2001): A solution i wins a tournament with another solution j (denoted as $i \prec_c j$) if solution i has a better rank ($i_{rank} < j_{rank}$) or i and j has the same rank but solution i has a better crowding distance than solution j ($i_{rank} = j_{rank}$ and $i_{distance} > j_{distance}$). If i and j has the same rank and the same crowding distance then one of them is randomly chosen as a winner. $\prec_c$ is the crowded comparison operator and is formally defined as $i \prec_c j$ if ($i_{rank} < j_{rank}$) or ($i_{rank} = j_{rank}$ and $i_{distance} > j_{distance}$).

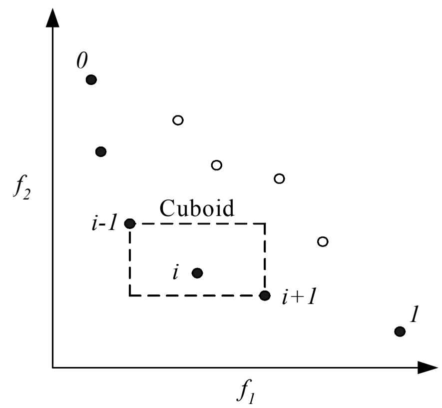


Figure 7: Illustration of the crowding distance calculation

Elitism: Elitism in NSGA-II is ensured by comparing the current population with previously found best non-dominated solutions (i.e. kept in a set $P^{'}$ as described above) and by combining the parent and child populations to form a combined population with size 2N. The combined population is then sorted according to non-domination. Solutions belonging to the best non-dominated front F_1 are of the best solutions in the combined population. If the size of F_1 is smaller than N, then all solutions in F_1 are selected for the new population. The remaining solutions of the new population are selected from subsequence non-dominated fronts in the order of their ranking F_2, F_3, and so on. This procedure is continued until N solutions are selected for the new population. To choose exactly N solutions, the solutions in the last front F_l are sorted using the crowded comparison operator ($\prec_c$) in descending order (crowding distance sorting), and the best solutions needed to fill N populations are chosen. The disadvantage of this elitism technique is that the non-dominated sorting is performed on a combined population of size 2N (parent and child populations), instead of size N like in most other MOEAs. Figure 8 illustrates the elitist mechanism of the NSGA-II.

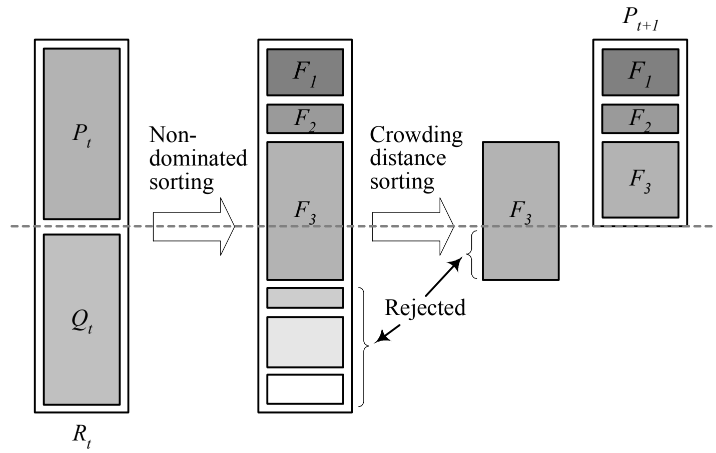


Figure 8: The elitist mechanism of NSGA-II

Real valued Representation: The NSGA-II supports real-valued representation, in which a chromosome is represented as a vector of decision variables in real-valued numbers, to remove weaknesses in binary representation. The binary representation traditionally used in simple GAs has some weaknesses as described in the following (Michalewicz, 1996; Deb, 2001). One of the weaknesses is that the so-called Hamming cliffs are formed when two numerically adjacent values have bit representations that are far apart. For example, the corresponding 8-bit binary representations for two decimal numbers 127 and 128 are 01111111 and 10000000. Despite being close in real values, these two binary strings have a Hamming distance of eight bits (the Hamming distance is the number of corresponding bits that differ). This presents a problem to a gradual search in the continuous search space. If, for example, the optimal solution is 127 and the current candidate solution is 128, 8 bits need to be changed to cause a small change in function evaluation. Another weakness is the inability to achieve any arbitrary precision in the obtained solutions when applied to high-precision numerical problems because the string length must be chosen a priori to enable the GAs to achieve a certain precision in the solutions. The higher the required precision, the longer is the string length. Goldberg, Deb, and Clark (1992) suggested a population size N, which is derived from the string length ℓ as $N = 1.65{*}2^{0.21\ell}$. Hence, the computational complexity of the algorithm is also increased. Since a fixed coding is used to code the decision variables, variable bounds must be such that they bracket the largest variable values. In many problems, this information is not usually a known priori. As a result, this may cause difficulty in using binary-coded representation for such problems.

Using real-valued representation, the NSGA-II can achieve arbitrary precision in the obtained solutions (the precision would depend on the underlying machine) and it can handle problems having a continuous search space easily (because there is no Hamming cliffs issue) when compared to binary-coded GAs. Moreover, the real-valued representation of the solutions is close to the natural formulation of many optimization problems. There are no differences between the genotype (coding) and the phenotype (search space). Hence, the coding and decoding processes that are needed in the binary-coded GAs are avoided, thus increasing the algorithm's speed. However, in real-coded GAs (GAs that are based on real-valued representation), the main challenge is how to create a new pair of offspring vectors from a pair of real-valued parent vectors (crossover) or how to mutate a real-valued variable vector that simulates the way mutation work in binary-coded GAs. Regarding this challenge, the NSGA-II uses the simulated binary crossover (SBX) operator (Deb & Agrawal, 1995) and the polynomial mutation operator (Deb & Goyal, 1996) to apply crossover and mutation directly to real-valued decision variables respectively.

The following equations illustrate the working mechanism of a SBX operator for creating two offspring ($x_i^{(1,t+1)}$ and $x_i^{(2,t+1)}$) from two parent solutions ($x_i^{(1,t)}$ and $x_i^{(2,t)}$) at generation *t* (Deb, 2001):

$$x_i^{(1,t+1)} = 0.5\left[\left(1+\beta_{q_i}\right)x_i^{(1,t)} + \left(1-\beta_{q_i}\right)x_i^{(2,t)}\right], \tag{10}$$

$$x_i^{(2,t+1)} = 0.5\left[\left(1-\beta_{q_i}\right)x_i^{(1,t)} + \left(1+\beta_{q_i}\right)x_i^{(2,t)}\right]. \tag{11}$$

$$\beta_{q_i} = \begin{cases} \left(2r_i\right)^{\frac{1}{\eta c+1}}, & if\ r_i \le 0.5; \\ \left(\frac{1}{2(1-r_i)}\right)^{\frac{1}{\eta c+1}}, & otherwise. \end{cases} \tag{12}$$

In the above equation, β_{q_i} is a spread factor, r_i is a random number $\in$ [0,1], and the distribution index η_c is any non-negative real number that a user can set. Thus, the SBX operator introduces an additional user-defined parameter η_c. This parameter affects the probability distribution of the SBX operator.

The following equations describe how the polynomial mutation operator is used to apply mutation directly to real-valued decision variables (Deb & Goyal, 1996). Let x_i be the value of the *i-th* variable. The result of the mutation on x_i is the new value y_i obtained by the following equations (Deb, 2001):

$$y_i^{(1,t+1)} = x_i^{(1,t+1)} + (x_i^{(U)} - x_i^{(L)})\bar{\delta}_i \tag{13}$$

$$\bar{\delta}_i = \begin{cases} (2r_i)^{\frac{1}{(\eta m+1)}} - 1, & if\ r_i \le 0.5, \\ 1-[2(1-r_i)]^{\frac{1}{(\eta m+1)}}, & if\ r_i > 0.5. \end{cases} \tag{14}$$

In the above equation, t is the generation counter, $x_i^{(U)}$ and $x_i^{(L)}$ are lower and upper bound of x_i respectively, r_i is a random number $\in [0,1]$, the distribution index η_m is any non-negative real number that a user can set and it directly controls the shape of the probability distribution. The polynomial mutation operator, however, introduces an additional user-defined parameter η_m.

Parameter Control Techniques in MOEAs

Self-adaptive mutation parameter: Laumanns et al. (2001) investigated whether parameter control techniques for mutation rate used in single-objective GAs can be used on MOEAs and what modifications are required to make these techniques work for multi-objective cases. This investigation has a significant contribution to the parameter control research in MOEAs by showing that self-adaptive mutating step size techniques used in single-objective ESs works differently in the multi-objective cases. Their study focused on the problem of convergence to the Pareto-optimal set. Laumanns et al. pointed out that research focusing on the role of parameter control in MOEAs remains rare. Most MOEAs such as NSGA-II, PAES, and SPEA2 support static parameters, where the parameter settings are initialized at the beginning of a MOEA's execution and fixed during the course of its execution. They emphasized that when large search spaces are to be explored in solving complex optimization problems, adaptive variation parameters are mandatory to achieve both a satisfactory rate of progress towards the Pareto-optimal front and a Pareto-optimal solution set because the proper parameter settings in the earlier stages of a GA's run usually become less efficient during the later stages. One of these parameters is the mutation rate: With too strong variation, the evolution becomes a pure random search; with too weak variation, no real progress can be achieved. The small region of appropriate mutation rate (known as the 'evolution window') depends on the landscape of the objective function, which is usually is unknown a priori. Hence, adaptation mechanisms for the mutation rate are a necessity for many optimization problems. Two types of functions whose components are well studied in the context of mutation control in single objective optimization are used: the sphere model and a multi-modal (with more than one solutions) function. The sphere model is a uni-modal function, which has been used as a reference in many theoretical studies of EAs and especially in the context of self-adaptive Evolution Strategies (ES). These two functions are listed below:

$$F(x) = \begin{pmatrix} (x-c_1)^2 \\ \dots \\ (x-c_m)^2 \end{pmatrix}, \qquad G(x) = \begin{pmatrix} \sum_{i=1}^{n-1} (-10e^{-0.2\sqrt{x_i^2 + x_{i+1}^2}}) \\ \sum_{i=1}^{n} (|x_i|^{0.8} + \sin^3(x_i)) \end{pmatrix} \tag{15}$$

In the multi-sphere model F function, the authors restricted to m = 2 (for 2 objectives) with $c_1 = (1,0,0,\ldots,0)$ and $c_2 = (0,1,0,\ldots,0)$. This function is a simple multi-objective test function for infinite search spaces because it is continuous, non multi-frontal, and the Pareto set can be determined analytically. However, since the search space is infinite, convergence depends crucially on the starting point and the mutation rate. The G function is part of van Veldhuizen's test function suite (1999) for MOEAs and features a disconnected Pareto sets. Floating point representation is used for the decision variables. An individual consists of n decision variables and one mutation step size. Thus, a single mutation step size is used for all variables. Mutation is carried out by first multiplying the step size with an instantiation of a log-normally distributed random variable: $\sigma^{(t+1)} = \sigma^{(t)} expN(0, \tau_0)$, where $\tau_0 = 1/\sqrt{n}$ and t is a generation counter.

Thereafter, a normal-distributed random vector with zero mean and variance $(\sigma^{(t+1)})^2$ is added to the individual: $x^{(t+1)} = x^{(t)} + N(0,(\sigma^{(t+1)})^2)$.

The self-adaptive mutation step size is applied to the standard $(\mu,\kappa,\lambda,\rho)$ ES where μ denotes the number of possible parents, κ the maximum lifespan of individuals (measured in generation), λ the number of offspring individuals, and ρ the number of crossover partners. A solution is assigned a fitness value equal to its non-dominated level within the correspondence population. For the single-objective case of the sphere model (first component of *F*), it has been known that self-adaptive ESs exhibit linear order convergence. However, the result shows that for the multi-objective case (the multi-sphere model), this behavior is valid only for the solutions, which are far away from the Pareto-optimal set. After a period of exponential decreasing the distance of the solutions to the Pareto-optimal set, the solutions suddenly start to oscillate around a small but fixed final distance to the Pareto-optimal set. It has been observed that when the solutions converge near the Pareto-optimal set, not only the success rate decreases but also the normalized average progress for successful mutations. As a result, it is increasingly difficult for the solutions to converge closer to the Pareto-optimal set. The same behavior exhibits on the multi-modal function *G*. Based on the experiments, Laumanns et al. concluded that self-adaptive Pareto-based ESs have difficulties to converge to the Pareto-optimal set due to the slow selection pressure of the ranking selection method based on the concept of Pareto optimality. They suggested a possible way to improve the convergence properties of self-adaptive MOEAs is to incorporate elitism to prevent possible divergence caused by increasing mutation step sizes and low selection pressure.

Self-adaptive crossover and mutation parameters: Abbass (2002) implemented self-adaptive crossover and mutation rates into the Pareto Differential Evolution (PDE) algorithm (Hussein A. Abbass, Sarkar, & Newton, 2001) and named the modified version as SPDE (Self-adaptive Pareto Differential Evolution). PDE is a multi-objective adaptation of the original Differential Evolution (DE) algorithm introduced by Rainer Storn and Kenneth Price (1996) for optimization problems over continuous domains. The DE algorithm is a population-based algorithm like GAs using the similar operators: crossover, mutation and selection. The main difference is that all solutions in the population have the same chance of being selected as parents without dependence of their fitness value. Another different is that in reproduction GAs rely on crossover while DE relies on mutation operation. However, the techniques for self-adaptive crossover and mutation rates used in SPDE are worth studying because Abbass's goal for the SPDE is also to make the algorithm easy to use and more attractive for decision makers in real life applications with regard to the parameter tuning. In the SPDE, both crossover and mutation rates are inherited from parents in the same way crossover is carried out for the decision variables. The SPDE works as follows. An initial population is generated randomly from a Gaussian distribution with mean 0.5 and standard deviation 0.15 (G(0.5, 0, 1.5)). Crossover and mutation rates are initialized in the initial population from a uniform distribution between [0, 1]. The main loop is listed in the following steps:

Step 1: Evaluate every solution in the population for non-dominance.

Step 2: Remove all dominated solutions the population.

Step 3: If the number of non-dominated solutions in the population > the user's specified maximum, do the following two steps to maintain diversity in the solutions:

- Apply the neighborhood distance function, which is the average Euclidean distance between the closest two points, to the non-dominated solutions:

$$D(x) = (\min \| x - x^i \| + \min \| x - x^j \|)/2 \quad \text{where } x \neq x^i \neq x^j$$

- Remove non-dominated solution with the smallest distance from the population until number of non-dominated solutions ≤ allowed maximum.

Step 4: Perform reproduction on the remaining non-dominated solutions as follows:

- Select three solutions at random – one as a main parent (α_1) and two as supporting parents ($\alpha_2,\ \alpha_3$). Select a variable *j* at random.
- Self-adaptive crossover rate is $x_c^{child} \leftarrow x_c^{\alpha_1} + G(0,1) \times (x_c^{\alpha_2} - x_c^{\alpha_3})$
- Self-adaptive mutation rate is $x_m^{child} \leftarrow x_m^{\alpha_1} + G(0,1) \times (x_m^{\alpha_2} - x_m^{\alpha_3})$
- Apply crossover: Each variable *i* in the main parent ($x_i^{\alpha_1}$) is perturbed by adding to it a ratio, $F \in$ Gaussian(0,1), of the difference between the two values of this variable in the two supporting parents as follows:

Table 4: Pseudo-code for a crossover operation in SPDE

```
For each variable x_i^{α1} ∈ α1  Do
  If uniform-probability(0,1) > x_c^{child} or i = j
    x_i^{child} ← x_i^{α1} + G(0,1) × (x_i^{α2} − x_i^{α3})
  Else
    x_i^{child} ← x_i^{α1}
End For
```

- Apply mutation: if the child solution dominates the main parent then apply mutation and place the child solution into the population as follow:

Table 5: Pseudo-code for a mutation operation in SPDE

```
If child solution dominates the main parent (x^{child} ≺ α1)
  If uniform-probability(0,1) > x_m^{child}
```

```
    For each variable $x_i^{\alpha_1} \in \alpha_1$ Do
        $x_i^{child} \leftarrow x_i^{child} + G(0,0.1) \times range$
        ;range is the difference between the maximum and
        ;the minimum value that the variable can take
    End For
  End If
  Place the child solution into the population
End If
```

- The reproduction process continues until the population is filled with N non-dominated solutions.

Step 5: The process continues until the terminate conditions are satisfied.

The SPDE was tested on four benchmark problems used in (Zitzler & Thiele, 1999). The solutions obtained by the SPDE were compared with the solutions of thirteen other MOEAs including the following popular MOEAs: Vector Evaluated GA (VEGA) (Schaffer, 1985), Non-dominated Sorting GA (NSGA) (N. Srinivas & Kalyanmoy Deb, 1994), Niched Pareto GA (NPGA) (Horn et al., 1994), Strength Pareto Evolutionary Algorithm (SPEA) (Zitzler & Thiele, 1998), Pareto-Archived Evolution Strategy (PAES) (Knowles & Corne, 1999). The results showed that the SPDE outperformed some of the MOEAs.

Adaptive population size: Devireddy and Reed (2004) incorporated the adaptive population size technique (similar to the multiple population approach used in the parameter-less GA (Harik & Lobo, 1999)) and the ε-dominance archiving technique (described in Step 2 below) used in the ε-MOEA (Deb, Mohan, & Mishra, 2003) into the NSGA-II. They named the modified version as ε-NSGA-II. The parameter-less GA has been described in the sub-section “Parameter Control Methods in Simple GAs” above. The ε-MOEA was developed by Deb, Mohan, & Mishra based on the ε-dominance concept (a solution x_a in X is said to ε-dominate a solution x_b in X (denoted as $x_a \prec_\varepsilon x_b$) for some $\varepsilon > 0$ if $(1+\varepsilon)f_i(x_a) \le f_i(x_b)$ for all $i = 1,\ldots,k$.) (Laumanns et al., 2002). The ε-dominance concept requires the user to define the precision with which they want to evaluate each objective by specifying an appropriate ε value for each objective. The ε-NSGA-II uses the ε values to find an approximation of Pareto-optimal set or ε-approximate Pareto-optimal set that meets the user-defined precisions. There are three main steps in the ε-NSGA-II:

Step 1: A MOP is solved using the original NSGA-II with fixed crossover and mutation probabilities (p_c = 0.9, p_m = 0.5) as suggested by Deb (Deb, 2001). The initial population size is set arbitrarily small (i.e. N = 5) so the NSGA-II can search for non-dominated solutions with a low computational cost. Coello and Pulido (2001) have shown that for some problems, a MOEA can find approximate Pareto-optimal solutions effectively using small population sizes.

Step 2: Similar to the ε-MOEA, the ε-NSGA-II maintains two co-evolving populations: an EA population $P(t)$ and an archive population $A(t)$ (where t is the iteration counter). Initially, the archive population $A(0)$ is initialized with the non-dominated solutions of $P(0)$ generated by the NSGA-II. Thereafter, the archive population $A(t)$ is updated as described in the following (Deb et al., 2003). Every solution in the archive is assigned an identification array ($B = (B_1, B_2, \ldots, B_m)^T$, where m is the number of objectives) as follows:

$$B_i(f) = \begin{cases} \lfloor (f_i - f_i^{\min})/\varepsilon_i \rfloor, & \text{for minimizing } f_i \\ \lceil (f_i - f_i^{\min})/\varepsilon_i \rceil, & \text{for maximizing } f_i \end{cases} \tag{16}$$

In the above equation, $f_i^{\min}$ is the minimum possible value of the *i*-th objective and ε_i is the allowable precision in the *i*-th objective below which two values are unacceptable to the user. The identification array divides the entire objective space into hyper-boxes, each have ε_i in the *i*-th objective. A solution *p* is selected at random from *P*(*t*), which contains non-dominated solutions generated in generation *t* of the NSGA-II run. An identification array is calculated for the solution *p*. The solution *p* is then compared with each solution in *A*(*t*) for ε-dominance. If the solution *p* is ε-dominated by any archive solution, *p* is not accepted. On the other hand, if the solution *p* ε-dominates any archive solution, *p* replaces the dominated archive solution. If neither of these two cases occurs, then it means that *p* is ε-non-dominated with the archive solutions. This case is separated into two. If the ε-non-dominated solution *p* shares the same hyper-box (the same *B* vector) with an archive solution, then two solutions are checked for the usual non-domination. Whichever solution dominates other solutions or is closer to the *B* vector (in term of the Euclidean distance) is retained. If the ε-non-dominated solution *p* does not share the hyper-box with any archive solution, *p* is accepted. Thus, the algorithm maintains the diversity in the archive by allowing only one solution to occupy each pre-assigned hyper-box on the Pareto-optimal front. Moreover, with this ε-dominance archiving technique, no specific upper limit on the archive size needs to be set. The archive size can be calculated from the chosen ε-vector as shown in (Laumanns et al., 2002). For example, the archive size can be calculated for a multiplicative ε as:

$$|F_\varepsilon| \le \left(\log K / \log(1+\varepsilon)\right)^{m-1} \tag{17}$$

where F_ε is ε-approximate Pareto-optimal set; $1 \le f_i \le K$, $K \ge \varepsilon_i$ for all $i \in \{1, \ldots, m\}$; and m is the number of objectives.

Step 3: This step checks if the user-specified performance and termination criteria are satisfied and the Pareto optimal set has been sufficiently quantified (as defined in equation (17)). The user-specified performance is checked by determining the distance of the obtained non-dominated set to a known Pareto-optimal set for the test problem given the specified tolerance values (ε values for each objective). The termination criteria include the maximum acceptable run time or the minimum percentage change in the number of non-dominated solutions for two successive runs to be considered identical. If the criteria are not satisfied, the population size is doubled and the search is continued. When doubling the population, the initial population of the new run has solutions injected from the archive at the end of the previous run.

The ε-NSGA-II was tested using a suite of two-objective test problems that are popular in literature including the test problems ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6 listed in Table 32 of Appendix A. The performance of the ε-NSGA-II is measured using the same convergence metric used to measure the performance of ε-MOEA (Deb et al., 2003). The results show that the ε-NSGA-II requires at least 60% fewer function evaluations than the original ε-MOEA. The ε-NSGA-II removes the population size parameter by making it adaptable, but users have to define the precision with which they want to evaluate each objective by specifying an appropriate ε value for each objective. Therefore, the ε-NSGA-II introduces a new parameter: the ε-vector.

Running Performance Metrics

In order to adjust values of the parameters of a MOEA during its run, the progress of a MOEA run must be monitored and evaluated, which involves comparing non-dominated solution sets among generations to see how the obtained solutions vary with generations. Performance metrics can be integrated into a MOEA to measure the quality of the obtained solutions during its run in order to provide progress information for adjusting values of the parameters. Unlike in single-objective optimization, where the performance metric is directly related to the objective function being optimized (for both being scalar quantities), in multi-objective optimization, two primary functionalities must be achieved: (i) approximating the Pareto-optimal front and (ii) maintaining a diverse set of solutions. Based on these two functionalities, two performance metrics can be devised (Deb & Jain, 2002): (i) a convergent metric for measuring the convergence of solutions to the Pareto-optimal front and (ii) a diversity metric for measuring the diversity of solutions. Such a set of two metrics enables two or more non-dominated solution sets to be compared among each other in terms of their functional performances.

There are interests in measuring different things in MOEAs. One may be interested in having a robust MOEA that approximates the global Pareto-optimal front of a problem consistently, rather than a MOEA that converges to the global Pareto-optimal front but only occasionally. This performance measure is usually applied to the final non-dominated set obtained by a MOEA. One may also be interested in generation wise performance measure for analyzing the behavior of an MOEA during the run in order to measure its capabilities to approximate progressively a non-dominated solution set to the global Pareto-optimal front of the problem being solved (assuming the true Pareto-optimal front is known) and to maintain diversity. This generation wise performance measure requires running metrics, which have the following properties (Deb & Jain, 2002):

- The metric should take a value between zero and one in an absolute sense. Since the metric is to be compared generation wise, an absolute scaling of a running metric between zero and one will allow to assess the change of the metric value from one generation to another.
- The target (or desired) metric value (computed for an ideally converged and diversified set of solutions) must be known.
- The metric should provide a monotonic increase or decrease in its value, as the population is improved or deteriorated slightly. This will also help in evaluating the extent of superiority of one approximation set with another.
- The metric should be scalable to any number of objectives. Although this is not a necessary property, but if followed, it will certainly be convenient for evaluating scalability issues of MOEAs in terms of number of objectives.
- The metric should be computationally efficient.

Several performance metrics have been suggested in the MOEA literature. Most of them are applied to the final non-dominated set obtained by a MOEA to evaluate its performance and may not be efficient to be used as running performance metrics (Deb & Jain, 2002). For example, the $D1_R$ metric (average distance of reference points from the approximate set) suggested by Czyzak and Jaszkiewicz (1998) for convergence measure, or the *S*-measure (used in (Zitzler, 1999)) for diversity measure cannot be adequately used due to computational expenses of the *R*-metrics calculation. The running convergent metric and diversity metric, which are to be investigated in this dissertation, are described in the following.

Running Metric for Convergence: The dissertation investigates the running convergence metric suggested by Deb & Jain (2002) because the metric can work with an unknown set of Pareto-optimal solutions (the Pareto-optimal set is usually unknown in advance for real-world problems) and it is efficient. In this metric, a reference set P^* can be either a set of Pareto-optimal solutions (if known) or an agglomeration of populations. An agglomeration of populations may be obtained in the following way. First, a MOEA is run for T generations and the generation wise populations (P_t, $t = 0, 1, \ldots, T$) are stored. Thereafter, the non-dominated solutions F_t of each population are combined together and the reference set is defined as the non-dominated set of the combined populations: $P^* =$ *non-dominated of* $(\bigcup_{t=0}^{T} F_t)$. The convergence metric for a population P at generation t (denoted as P_t) is calculated in the following steps (Deb & Jain, 2002):

Step 1: Identify the non-dominated set F_t of P_t.

Step 2: From each point i in F_t, calculate the smallest normalized Euclidean distance to P^* as follows:

$$d_i = \min_{j=1}^{|P^*|} \sqrt{\sum_{k=1}^{m} \left(\frac{f_k(i) - f_k(j)}{f_k^{\max} - f_k^{\min}} \right)^2} \tag{18}$$

Here, m is the number of objective functions; $f_k^{\max}$ and $f_k^{\min}$ are the maximum and the minimum function values of *k-th* objective function in P^*.

Step 3: Calculate the convergence metric by averaging the normalized distance for all points in F_t:

$$C(P_t) = \frac{\sum_{i=1}^{|F_t|} d_i}{|F_t|}. \tag{19}$$

In order to keep the convergence metric within [0,1], once the above metric values are calculated for all generations, the values of $C(P_t)$ is normalized by its maximum value (usually $C(P_0)$):

$$\bar{C}(P_t) = \frac{C(P_t)}{C(P_0)} \tag{20}$$

A small convergent metric value indicates the obtained solution set is close to the Pareto-optimal front.

The convergent metric above has been applied on solutions obtained using NSGA-II on four two-objective test problems suggested in literature: ZDT1, ZDT2, ZDT3, and ZDT6 (see Table 32 in Appendix A) and one real-world two-objective gearbox design problem. The published results demonstrate that non-dominated sets progressively approach the reference set (using known Pareto-optimal front) for all test problems (Deb & Jain, 2002). Using the population-agglomeration technique, the convergent metric above has also been applied on solutions obtained using NSGA-II on two three-objective test problems DTLZ2 and DTLZ5 borrowed from Deb, Thiele, Laumanns, & Zitzler (2002). For the test problem DTLZ2, which has a spherical Pareto-optimal surface, the convergent metric values suggest that several non-dominated solutions even at the final generation have not approximated to the true Pareto-optimal set. For the test problem DTLZ5, which has a two-dimensional Pareto-optimal curve, the convergent metric values show that non-dominated solutions approximate to the true Pareto-optimal set.

Deb and Jain (2002) did not evaluate the population-agglomeration technique on the same test problems (ZDT1, ZDT2, ZDT3, and ZDT6) that were used to evaluate the running convergent metric using known Pareto-optimal front. Therefore, the effectiveness of the population-agglomeration technique is not validated.

Running Metric for Diversity: The dissertation investigates the running metric for measuring the diversity of the obtained solutions suggested by Deb et al. (2002). This metric is defined as follows:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} \left| d_i - \bar{d} \right|}{d_f + d_l + (N-1)\bar{d}} \tag{21}$$

Here, the parameter d_f and d_l are the Euclidean distances between the known extreme solutions and the boundary solutions of the obtained non-dominated set. The parameter d_i is the Euclidean distance between two consecutive solutions of the obtained non-dominated set. The parameter $\bar{d}$ is the average of all Euclidean distance d_i, where i = 1, 2,…, (N – 1), assuming that there are N solutions on the first non-dominated front. For the most uniformly and widely spread out set of non-dominated solutions, the value of Δ is zero. For any other distribution, the value of the metric would be greater than zero.

The diversity metric above was used to measure the diversity of the non-dominated solution sets obtained by the NSGA-II (Deb, Pratap et al., 2002) on nine two-objective test problems suggested in literature (see Table 32 in Appendix A).

Summary of Known and Unknown

Several parameter control methods have been proposed and applied successfully for single objective optimization problems using simple GAs. One of the most significant empirical studies was performed by Bäck, Eiben, and van der Vaart (2000) in which simple GAs have one or all three parameters (p_m, p_c, and N) adjusted during the run. The results of this study show the superiority of the GA with adaptable parameters and suggest more studies on control techniques for adaptive population sizes. Research focusing on the role of parameter control in MOEAs remains rare. Most MOEAs such as NSGA-II, PAES, and SPEA2 support static parameters, where the parameter settings are initialized at the beginning of a MOEA's run and fixed during the course of its run. The use of fixed parameter settings may lead to slow convergence and sub-optimal obtained solutions, especially when large search spaces are to be explored in solving complex optimization problems because the proper parameter values are not fixed but varied during a run. Some previous studies have applied parameter control techniques to MOEAs. However, these studies focus on one or two parameters in isolation and ignore other parameters. These studies have shown that parameter control techniques used in single-objective GA work differently in the multi-objective cases (Laumanns et al., 2001; Tran, 2005). In contrast to single-objective optimization, where objective function and fitness function are often the same, in multi-objective optimization, both fitness assignment and selection must support several objectives. The result of the multi-objective optimization process is usually not a single solutions but a set of trade-off solutions. These trade-off solutions converge towards different areas of the Pareto-optimal front and proper parameter values differ between these solutions. Therefore, self-adaptation of parameter values reveals an additional difficulty compared to single objective optimization. Moreover, in a MOEA, each solution is assigned a fitness value equal to its non-dominated rank in the population (1 is the best rank, 2 is the next best rank and so on), which is determined by using a non-dominated sorting procedure. Thus, solutions that have the best rank values (rank 1) constitute the first non-dominated front. This fitness assignment imposes a barrier in comparing two different non-dominated solutions set. In order to adjust the values of parameters, the progress of a MOEA run must be monitored and evaluated, which involves comparing solution sets among generations to see how the obtained solutions vary with generations. However, all of these solutions are in the first non-dominated front and have the same rank value. As a result, it is difficult to determine the better non-dominated solution set between two sets of non-dominated solutions. Performance metrics can be integrated into a MOEA to measure the convergence and diversity of the obtained solutions during its run in order to monitor and provide the MOEA's progress for adjusting the values of parameters. These running performance metrics should be reliable and efficient in order to provide correct progress information without spending too much time on metric calculations and taking away time for finding the solutions. Several performance metrics have been introduced in the MOEA literature. However, most of these metrics are applicable to two-objective problems (Deb & Jain, 2002). The convergent metric and diversity metric investigated in this study also have not been applied to measure the quality of solutions on problems with more than three objectives. Thus, many works remain to be done on parameter control techniques for MOEAs.

Contributions

This dissertation advances knowledge and makes a significant contribution to the MOEA field by automating the process of selecting appropriate MOEA parameter values, and making MOEAs easier to use and available to more users. In addition, this dissertation improves efficiency of a MOEA because when large search spaces are to be explored, adaptive variation parameters are necessary in order to achieve both a satisfactory rate of progress towards the Pareto-optimal front and a Pareto-optimal solution set.

Chapter 3

Methodology

This chapter is organized as follows: First, it describes the research method used in carrying out this study. Second, it presents the specific procedures that are used to develop the algorithm ANSGA-II. Third, it continues with a discussion of how the results of this study are presented along with an explanation of the resource requirements to complete this dissertation. Finally, a discussion of the reliability and validity of results are provided along with a summary of the chapter.

Research Methods Employed

The evaluation research method is used as an approach in this dissertation. In evaluation research, researchers perform formative studies while a new product is being developed, then summative studies when the product has been completed (Glatthorn, 1998). This research consists of the following major steps:

- Perform formative studies of existing parameter control techniques in simple GAs and MOEAs. The results of these studies identify the available techniques that can be used, issues and barriers that are needed to be resolved. These studies have been described in Chapter 2 above.
- Perform formative study of the NSGA-II. The result of this study ensures the successful development of the ANSGA-II.
- Develop the new algorithm ANSGA-II based on the NSGA-II.

- Evaluate the ANSGA-II against the original NSGA-II using the same benchmark test problems that were used in the study of the original NSGA-II. Since the same test problems are used, the results generated by the ANSGA-II can be easily compared to those of the NSGA-II for validation.
- Perform summative studies, which include validating and presenting the results.

Specific Procedures to be employed

This research involves building on NSGA-II to develop the proposed algorithm ANSGA-II and to evaluate the ANSGA-II using benchmark test problems. In this section, the NSGA-II is described then the steps to develop the ANSGA-II are outlined.

Description of the NSGA-II

The NSGA-II consists of three new techniques: a fast non-dominated sorting procedure, a crowded distance assignment procedure, and a crowded comparison operator. The pseudo codes for these three techniques are presented next. Thereafter, the NSGA-II is outlined.

Fast Non-Dominated Sorting Procedure: As described in the literature review for the NSGA-II above, the procedure for finding non-dominated front used in NSGA-II is similar to the non-dominated sorting procedure suggested by Goldberg (1989) except that a better bookkeeping strategy is used to make it more efficient. Pseudo-code for the find non-dominated front procedure is presented in Table 6 (Deb, Pratap et al., 2002).

Table 6: Pseudo-code for the procedure FindNonDominatedFront

Procedure FindNonDominatedFront(P)	
$P' = P[0]$	*;include first solution in P*
For each $p \in P$ *and* $p \notin P'$ *Do*	*;process one solution at a time*
$P' = P' \cup \{p\}$	*;include p in* P' *temporarily*
For each $q \in P'$ *and* $q \neq p$ *Do*	*;compare p with other solutions in* P'

If p ≺ q then P' = P' \ {q} ;*if p dominates a member of P', delete it*
Else If q ≺ p then P' = P' \ {p} ;*if p is dominated by other member of P',*
End For ;*remove it from P'*
End For

To find other fronts, the non-dominated solutions found in P' will be discounted from P and the above procedure is repeated. Pseudo-code for the fast non-dominated sorting procedure is presented in Table 7 (Deb, Pratap et al., 2002).

Table 7: Pseudo-code for the procedure FastNonDominatedSorting

```
Procedure FastNonDominatedSorting(P)
i = 1                                ;i is the front counter
Repeat
   Fi = FindNonDominatedFront(P)     ; find the non-dominated front i
   P = P \ Fi                        ;remove non-dominated solutions from P
   i = i + 1                         ;increment the front counter
Until all the solutions in the population P are ranked
```

Crowded Distance Assignment Procedure: As described in the literature review for the NSGA-II above, to maximize the diversity of the obtained solutions, the NSGA-II uses the crowded comparison approach. This approach requires every solution i in the population has two attributes: a non-domination rank (i_{rank}) and a crowding distance ($i_{distance}$). The value of i_{rank} is obtained through the fast non-dominated sort as described above. The crowding distance $i_{distance}$ of a solution i is a measure of the perimeter of the largest cuboid enclosing the solutions i, without including any other solution in the population, formed by using the nearest neighbor solutions as the vertices as illustrated in Figure 7 above. The process of assigning crowding distance ($i_{distance}$) values to all solutions in the population requires the population sorted according to each objective function value in their ascending order of magnitude. Thereafter, for each objective function, the boundary solutions (solutions with smallest and largest function values) are assigned an infinite distance value (e.g. $1e^{14}$). All other intermediate solutions are assigned a distance value equal to the absolute difference in the function values of two adjacent solutions. This calculation is repeated with other objective functions. The overall crowding distance value is calculated as the sum of individual distance values corresponding to each objective. Pseudo-code for the crowding distance assignment procedure is presented in Table 8 (Deb, Pratap et al., 2002). In the code, *I[i+1].m* refers to the *m*-th objective function value of the *i*-th individual in the set *I*. After all solutions in the set *I* are assigned a crowding distance values, solutions can be compared for their extent of proximity with other solutions. A solution with a smaller crowding distance value is more crowded by other solutions.

Table 8: Pseudo-code for the procedure CrowdingDistanceAssignment

```
Procedure CrowdingDistanceAssignment(I)
ℓ = |I|                              ;number of solutions in I
For each solution i ∈ I
   set I[i]_distance = 0             ; initialize distance
End For
For each objective m
   I = sort(I, m)                    ;sort using each objective value in ascending order
   I[1]_distance = I[ℓ]_distance = 1e^14   ;so that boundary solutions are always selected
   For i = 2 to (ℓ-1) Do             ;for all other solutions
      I[i]_distance = I[i]_distance + (I[i+1].m – I[i-1].m)
   End For
End For
```

Crowded Comparison Operator: Every solution i in the population has two attributes: a non-domination rank (i_{rank}) and a crowding distance ($i_{distance}$). The values of i_{rank} and $i_{distance}$ are obtained through the fast non-dominated sort procedure and the crowding distance assignment procedures respectively as described above. The crowded comparison operator ($\prec_c$) is formally defined as:

$$i \prec_c j \ \ if \ \ (i_{rank} < j_{rank}) \ or \ (i_{rank} = j_{rank} \ \ and \ \ i_{distance} > j_{distance}) \tag{22}$$

The crowded comparison operator guides the selection process in NSGA-II (known as crowded tournament selection) towards a uniformly spread-out Pareto-optimal front. The crowded tournament selection is defined as follows (Deb, 2001): a solution *i* wins a tournament with another solution *j* if $i \prec_c j$. If *i* and *j* has the same rank and the same crowding distance then one of them is randomly chosen as a winner. Pseudo-code for the NSGA-II selection procedure is presented in Table 9.

Table 9: Pseudo-code for the procedure Selection

```
global N = 100                ;population size
Procedure Selection(Pt, Mt)
n = 0 and Mt = 0    ;empty
Repeat
   p1 =  select first solution from Pt at random
   p2 = select second solution from Pt at random
   If p1 ≺c p2
       Mt_n = p1            ;n-th solution in Mt = p1
   Else If p2 ≺c p1
       Mt_n = p2            ;n-th solution in Mt = p2
   Else
       Mt_n = pick p1 or p2 randomly
   End If
   n = n + 1
Until n = population size N
Return Mt
```

Outline of the NSGA-II: The pseudo-code for the NSGA-II is listed in Table 10, Table 11, and Table 12. Thereafter, the algorithm is described in the form of outline.

Table 10: Pseudo-code for the main procedure NSGA-II

```
global N = 100                          ;population size
Procedure NSGA-II()
t = 0;                                  ;initialize generation counter
Pt = random solutions for parent population P0
Pt = FastNonDominatedSorting(Pt)     ;F=(F1,F2,...), all non-dominated fronts of Pt
CrowdingDistanceAssignment(Pt)  ;calculate crowding distance in Pt
Repeat
   ;apply crowded tournament selection on Pt to create mating population Mt
   Mt = Selection(Pt, Mt);              ;see pseudo-code in Table 9 above
```

```
   ;apply SBX, and polynomial mutation to create new child population Q_t
   Q_t = SBX(M_t, Q_t);              ;see pseudo-code in Table 11 below
   Q_t = PolynomialMutation(Q_t)     ;see pseudo-code in Table 12 below
   ;the following code preserves elitism and diversity of solutions
   R_t = P_t ∪ Q_t                   ;combine parent and child population
   F = FastNonDominatedSorting(R_t) ;F=(F_1,F_2,...), all non-dominated fronts of R_t
   P_{t+1} = 0                       ;empty
   i = 1                             ;first front
   Repeat
      CrowdingDistanceAssignment(F_i)   ;calculate crowding distance in F_i
      P_{t+1} = P_{t+1} ∪ F_i    ;include i-th non-dominated front in the  parent population
      i = i + 1              ;check the next front for inclusion
   Until |P_{t+1}| + |F_i| > N Or No More F_i  ;until parent population is filled
   If |P_{t+1}| < N                          ;not fill up to N
     Sort(F_i, ≺_c)                          ;sort the last front in descending order using ≺_c
     P_{t+1} = P_{t+1} ∪ F_i[1 : (N - |P_{t+1}|)]   ;choose the first (N - |P_{t+1}|) solutions in F_i
   End If
   t = t + 1                                 ;increment generation counter
Until the terminate conditions are satisfied
```

Table 11: Pseudo-code for the procedure SBX

```
global N = 100      ;population size
global p_c = 0.9    ;crossover probability
global η_c = 20     ;distribution index for real-coded crossover
Procedure SBX (M_t, Q_t)
n = 0
Repeat
    random = random number ∈ [0,1]
    If random ≤ p_c              ;decide to do crossover or not
      For each variable i ∈ M_{t_n}  Do
         ;select two parents from mating pool M_t
         x_i^(1,t) = M_{t_n}^i          ;i-th variable of the n-th parent from M_t
         x_i^(2,t) = M_{t_{n+1}}^i      ;i-th variable of the (n+1)-th parent from M_t
         r_i = random number ∈ [0,1]
         If r_i ≤ 0.5
            β_{q_i} = (2r_i)^(1/(η_c+1))
         Else
            β_{q_i} = (1 / (2(1 − r_i)))^(1/(η_c+1))
         End If
         ;perform crossover on two parents (x_i^(1,t) and x_i^(2,t)) then store
```

```
        ;two offspring solutions in new population Q_t
        ; Q^i_{t_n} is i-th variable of the n-th solution in Q_t
        ; Q^i_{t_{n+1}} is i-th variable of the (n+1)-th solution in Q_t
        Q^i_{t_n} = 0.5[(1 + β_{q_i}) x_i^{(1,t)} + (1 − β_{q_i}) x_i^{(2,t)}]
        Q^i_{t_{n+1}} = 0.5[(1 − β_{q_i}) x_i^{(1,t)} + (1 + β_{q_i}) x_i^{(2,t)}]
      End For
    Else
      ;do not do crossover
      For each variable x_i ∈ P_{t_n} Do
         ;copy solutions from mating pool to new population
         Q^i_{t_n} = M^i_{t_n}         ; Q^i_{t_n} is i-th variable of the n-th solution in Q_t
         Q^i_{t_{n+1}} = M^i_{t_{n+1}} ; Q^i_{t_{n+1}} is i-th variable of the (n+1)-th solution in Q_t
      End For
    End If
    n = n + 2
Until n = (population size N)/2
Return Q_t
```

Table 12: Pseudo-code for the procedure PolynomialMutation

```
global N = 100     ;population size
global p_m = 0.5   ;mutation probability
global η_m = 100   ;distribution index for real-coded mutation
Procedure PolynomialMutation (Q_t)
n = 0
Repeat
   For each variable x_i ∈ Q_{t_n} Do
      r_i = random number ∈ [0,1]
      If r_i ≤ p_m            ;decide to do mutation or not
         If x_i > x_i^(L)     ; x_i^(L) is lower bound of x_i
            r_i = random number ∈ [0,1]
            If r_i ≤ 0.5
               δ̄_i = (2r_i)^(1/(η_m+1)) − 1
            Else
               δ̄_i = 1 − [2(1 − r_i)]^(1/(η_m+1))
            End If
            ; Q^i_{t_n} is i-th variable of the n-th solution in Q_t
            Q^i_{t_n} = x_i + (x_i^(U) − x_i^(L)) × δ̄_i    ;mutate i-th variable of x
         Else                 ; x_i = x_i^(L)
```

```
            ri = random number ∈ [0,1]
            ; Q^i_{t_n} is i-th variable of the n-th solution in Qt
            Q^i_{t_n} = ri × (xi^(U) − xi^(L)) + xi^(L)   ;mutate i-th variable of x
         End If
      End If
   End For
   n = n + 1
Until n = population size N
Return Qt
```

The NSGA-II procedure is described in the following steps:

Step 1: Five parameter settings: population size (N = 100), crossover probability (p_c = 0.9), distribution index for real-coded crossover (η_c = 20), mutation probability (p_m = 0.5), and distribution index for real-coded mutation (η_m = 100) are set by a user as suggested by Deb et al. (2002). These parameter values are fixed during the execution of the NSGA-II.

Step 2: Create a parent population P_0 of size N with random solutions.

Step 3: Sort population P_0 based on non-dominance, using the above procedure *FastNonDominatedSorting*.

Step 4: Calculate crowding distance in the initial population P_0.

Step 5: Apply crowded tournament selection to create a mating population M_t. Pseudo-code for the selection procedure is listed in Table 9. In this selection procedure, two solutions (p_1, p_2) from the parent population P_t are selected at random for a tournament using the crowded comparison operator as described previously (see equation (22)). The winner will be inserted in the mating pool for reproduction. This selection process is repeated until the mating pool is filled with N solutions.

Step 6: Apply crossover SBX on the solutions in the mating pool M_t to create a new population Q_t. Pseudo-code for the SBX procedure is listed in Table 11. For each solution in the mating pool the following steps are performed. The crossover probability (p_c) is compared with a random number in [0, 1] to determine if the crossover operation should be carried out or not on each solution. If the random number is less than or equal to the crossover probability then the SBX operator is applied on each variable of two parent solutions selected from the mating pool to produce two offspring solutions using equations (10), (11) and (12) as described in the literature review section for the NSGA-II above. These two offspring solutions are then stored in the new population. If the random number is greater than the crossover probability then the crossover operation is not carried out and the solution in the mating pool is simply copied over to the new population.

Step 7: Apply polynomial mutation on the solutions in the new population Q_t. Pseudo-code for the Polynomial Mutation procedure is listed in Table 12. For each variable in each solution of the new population the following steps are performed. The mutation probability (p_m) is compared with a random number in [0, 1] to determine if the mutation operation should be carried out or not on the current variable. If the random number is less than or equal to the mutation probability then the polynomial mutation operator is applied on the variable using equations (13) and (14) as described in the literature review section for the NSGA-II above.

Step 8: Combine parent and child populations to create a combined population ($R_t = P_t \cup Q_t$) of size $2N$.

Step 9: Sort R_t based on the non-domination (using the procedure FastNonDominatedSorting described above) and identify different fronts: F_i, $i = 1, 2, \ldots$, etc.

Step 10: Initialize new population P_{t+1} to empty ($P_{t+1} = 0$) and a counter $i = 1$.

Step 11: Repeat this step until up to N solutions are selected for the new population P_{t+1}. For each non-dominated front F_i, $i = 1, 2, \ldots$, etc. in R_t calculate crowding distances for solutions in F_i (using the procedure *CrowdingDistanceAssignment* described above) and select all solutions in F_i to fill P_{t+1}. Solutions belonging to the best non-dominated front F_1 are of the best solutions in the combined population R_t. If the size of F_1 is smaller than N, then all solutions in F_1 are selected for P_{t+1}. The remaining solutions of the new population P_{t+1} is selected from subsequence non-dominated fronts in the order of their ranking F_2, F_3, and so on.

Step 12: To select exactly N solutions for P_{t+1}, the solutions in the last front, which cannot fit into the new population P_{t+1}, are sorted using the crowded comparison operator ($\prec_c$) in descending order (crowding distance sorting), and the best solutions needed to fill N populations are chosen. Figure 8 in the literature review section for the NSGA-II illustrates the working mechanism of Step 8 to Step 12.

Step 13: Increment the generation-counter. If the terminate conditions are not satisfied go back to Step 5.

Development of the ANSGA-II

The specific steps to develop the ANSGA-II include:

- Implement the test problems: Nine un-constrained test problems with two objective functions, three constrained test problems with two objective functions, and one real-world problem with five objective functions and seven constraints are implemented in both the original NSGA-II and the ANSGA-II. These problems, which were used in the study of the original NSGA-II (Deb, Pratap et al., 2002) and for which Pareto-optimal solutions are known (except for the five-objective problem WATER), are listed in Appendix A. Each test problem is implemented as a procedure so it can be called and reused. For example, the test problem TNK listed in Table 33 of Appendix A is implemented as follows:

Table 13: Pseudo-code for the procedure TNK

```
Procedure TNK(f[], x[],cstr[])
;all functions are implemented as of minimization type
;negate maximization functions as necessary
;first fitness function
f[0] = x[0];
;second fitness Function
f[1] = x[1];
;constraints
cstr[0] = (float)(x[0]*x[0]+x[1]*x[1]-1.0-0.1*cos(16.0*atan(x[0]/x[1])));
cstr[1] = (float)((-square(x[0]-0.5) - square(x[1]-0.5) + 0.5)/0.5);
```

- Run the NSGA-II to solve each test problem one at a time: The obtained non-dominated solution sets are saved and used later to compare with those obtained by the improved ANSGA-II in order to evaluate the performance of the ANSGA-II. The obtained non-dominated solutions in objective space on two-objective test problems are plotted with two axes (*X* and *Y*) represent two objective functions. It is difficult to visualize a graph with more than 3-axes; therefore, the obtained non-dominated solutions on the five-objective WATER problem are plotted with the scatter-plot matrix method as suggested by Meisel (1993) and Cleveland (1994). The plotting graphs of the obtained non-dominated solutions on the test problems are compared visually with those published in the paper by Deb et al. (2002) to make sure that the test problems are implemented correctly.

- Incorporate the adaptable population size technique of into the ANSGA-II: This step ports already tested code from the parameter-less NSGA-II (Tran, 2005), which runs multiple populations simultaneously with various population sizes, into the ANSGA-II. In the parameter-less NSGA-II, the algorithm runs until it finishes with the last population with the maximum population size, or the user is happy with the obtained solutions and stops the program. In the ANSGA-II, the performance metrics are investigated for their effective use to determine the better non-dominated solution sets among two or more solution sets in order to stop running populations with inferior solution sets. Pseudo-code for the main procedure of the ANSGA-II is listed in Table 14.

Table 14: Pseudo-code for the main procedure ANSGA-II

```
;Main global constants and variables
INIT_POP_SIZE = 20
MAX_POP_SIZE = INIT_POP_SIZE * 40
CONVERGED_COUNT_THRESHOLD = 3

Individual : Record of
        rank                              ;rank of the individual
        real[MAX_VARIABLE]                ;list of real variables
        fitness[MAX_FUNCTIONS]            ;individual's fitness values of objective functions
        constr[MAX_CONSTRAINTS]           ;constraints values
        crowding_distance                 ;crowding distance of the individual
        ;the following four parameters are attached to each individual for co-evolution
        crossover_rate                    ;pc
        crossover_dix                     ;ηc
        mutation_rate                     ;pm,
        mutation_dix                      ;ηm
      End Record

P : Record of
        Size                              ;population size for this pop.
        MarkForDeletion                   ;when set to true, this pop. will be stopped running
        Converged                         ;true if this pop. has converged; o/w false
        ConvergedCnt                      ;count number of times this pop. has converged
        NoImprovementCnt                  ;count no improvement in performance metric value
        ConvergentMetricValue             ;convergent metric value on the obtained solutions
        DiversityMetricValue              ;diversity metric value on the obtained solutions
        NumGenRunEachTime                 ;specify number of gen. to run when its turn to run
        RunCount                          ;count number of runs
        MaxGeneration                     ;maximum number of generations for this population
        Solutions : individual[Size]      ;array of individuals as defined above
        Rank : Integer[Size]              ;number of individuals in each rank
```

```
 End Record

Procedure ANSGA-II ()
Pop : P[100]  ;array of 100 P
ix = 0            ;population index
current_pop_ix = 0
;1) the counter below enables a smaller population to run 4
;more generations than a larger one
counter : base 4 on 100
found_best_ever_solutions = FALSE
Repeat
   ;2) which population to run?
   If Not (PopConverged Or found_best_ever_solutions)
       ix = counter.IsCounterChange(ix)      ;change every 4 count
   End If
   If Pop[ix] does not exist.
      If Pop[current_pop_ix].Size < MAX_POP_SIZE    ;create new population?
         ;3) create new population with population size
         ;N =2 * current population size  and filled it with initialize its
         ;solutions, random crossover and mutation parameter values
         CreateNewPopulation(ix)     ;see Table 15 for pseudo-code of this proc.
      Else
         ;4) do not create new population
         counter.Reset ()
         ;continue running existing populations until convergence, or
         ;maximum number of function evaluations reached, or
         ;maximum number of generations reached
         Continue
      End If
   End If

   ;5) calculate metrics before yielding execution to another population
   If ix != current_pop_ix
       ;pseudo-code for proc. CalculatePerformanceMetric is listed in Table 16
       CalculatePerformanceMetric(current_pop_ix)
       current_pop_ix = ix     ;set current population
   End If

   ;6) run the NSGA-II that has been modified to support adaptable crossover and
   ;mutation parameters for number of generation specified in NumGenRunEachTime
   If Pop[ix] exists
       Pop[ix].NSGA-II ()
   End If

  ;7) has the population been converged?
  ;if there is enough non-dominated solutions in the first rank and the population
```

```
;has converged for at least number of converged count
PopConverged = Pop[ix].Rank[0] ≥ MinNumOfSolutions And
                  Pop[ix].ConvergedCnt ≥ CONVERGED_COUNT_THRESHOLD
If PopConverged
    found_best_ever_solutions = TRUE
   ;8) kill all other populations
   For all ii != ix do
     Kill Pop[ii]
   End For
;9) is the current population is close to convergence?
; see Table 19 for pseudo-code for IsPopulationCloseToConvergence
Else If IsPopulationCloseToConvergence(ix)
   ;10) if the current population waiting on further improvement
   ;stop running larger populations that are not close to convergence
   ;because it has worse solutions compared to the current population
   If Pop[ix].NoImprovementCnt ≥ 1 And Pop[ix].ConvergedCnt ≥ 1
     CountKillLargePop = 0
     For ii = ix+1 to Size of Pop
        If Pop[ii].ConvergedCnt = 0
           ;mark to stop running this larger population
          Pop[ii]. MarkForDeletion = true
          CountKillLargePop = CountKillLargePop + 1
        End If
     End For
     ;11) if  all larger populations are marked for deletion set
     ; found_best_ever_solutions = true to prevent creating new population
     If CountKillLargePop = Size of Pop – ix + 1
       found_best_ever_solutions = true
     End If
   End If
   ;12) if the current population has better solutions than smaller
   ;populations, stop running smaller populations
   For ii = 0 to ix-1
       If CompareTwoSolutionSets (ii, ix) = ix
          ;mark to stop running this smaller population
         Pop[ii]. MarkForDeletion = true
       End If
   End For
End If
;13) reach maximum number of generations allowed to run?
If Not PopConverged And Pop[ix].RunCount ≥ Pop[ix].MaxGeneration
  Pop[ix]. MarkForDeletion = true   ;stop running this population
  If Size of Pop = 1  ;is this the only population remained running
     ;14) something wrong, terminate the program
    break the loop
  End If
```

```
  End If
  ;15) now kill all the populations that have been marked for deletion
  For ii = 0 to Size of Pop – 1
    If Pop[ii]. MarkForDeletion = true
        Kill Pop[ii]
        ;NOTE: need to adjust indexes accordingly
    End If
  End For
   ;16) increase counter only solutions are not satisfied
   If NOT found_best_ever_solutions
     counter.Increase()
   End If
Until PopConverged  or  max. number of function evaluations reached
;17) calculate performance metric on the final non-dominated set
CalculatePerformanceMetric(current_pop_ix)
```

Similarly, to the simple parameter-less GA (Harik & Lobo, 1999), the ANSGA-II allows smaller populations more generations to run. The coordination of the array of populations is implemented with a counter base 4 (step 1). Overall, the *n*th population is allowed to run 4 times more generations than the *n*+1th population. If the next population to be ran has not yet been created, the algorithm creates a new population twice as large as the previous one (step 3) until the maximum population size (i.e. *MAX_POP_SIZE* = 800) is reached (step 4). Each time a population yields its execution (step 5), performance metric is calculated for the current population as described in pseudo-code for the procedure *CalculatePerformanceMetric* (see Table 16). The algorithm then calls the NSGA-II code to run the population for number of generations as specified in *NumGenRunEachTime*. When the ANSGA-II obtains enough diverse non-dominated solutions with good convergence, the algorithm continues running the population until its converged count reaches the threshold (step 7). Since the smaller populations have more generations (more time) to run, they expect to approximate to a diverse Pareto-optimal front faster than the larger one and the algorithm would stop running larger populations (step 8 and 10). On the other hand, if at any point in time, a larger population has a non-dominated solution set better than that of a smaller population then the algorithm would stop running the smaller population because the smaller population size is not large enough to commensurate the difficulty of the problem being solved (step 12). If a population is executed up to its allowed maximum number of generations but its non-dominated solution set is still not good, the algorithm would stop running it (step 13). If the solutions are not satisfied, the counter is increased and the algorithm continues (step 16). The algorithm continues until the diversity in the obtained non-dominated solutions is achieved or when the maximum number of function evaluations is reached. It then calculates the diversity metric on the final non-dominated solution sets before terminating the program (step 17).

- Implement the procedure CreateNewPopulation: Pseudo-code for the procedure *CreateNewPopulation* is listed in Table 15. The initial population is set to a small size (*INIT_POP_SIZE* = 20) because a MOEA can find approximate Pareto-optimal solutions for a small problem effectively with a small population size (Coello & Pulido, 2001). For problems with number of objectives greater than two, the initial population size is proportional with number of objectives (step 3) to commensurate the difficulty of the problem. The ANSGA-II supports a population with maximum size equals to 800 (*INIT_POP_SIZE* * 40). The value for *MinNumOfSolutions* is set to the initial population size. This value represents the minimum number of non-dominated solutions required in the first rank. For an initial population, the number of generations allowed to run when its turn to run is set to one (*NumGenRunEachTime* = 1). For a later population, it is created with a population size twice as large as the previous population size (step 6) and

NumGenRunEachTime is also set to double the value that of the previous population (step 7). This allows larger populations a little bit more generations to run instead of strictly adopting the counter base method (a smaller population is allowed to run four more generations than a larger one). The new population is then initialized with values for solutions, crossover parameters, and mutation parameters. If the population is a first one or it is a later but the previous population does not have good solutions, the population is initialized with random values (step 8 and 10). If it is a later population, the first half of the population is initialized with good solutions, crossover, and mutation values from the previous population (step 8 and 9). The second half is filled with random values (step 10). Step 11 in the procedure set the maximum number of generations that the population allowed to run (*MaxGeneration* = *NewPopSize* * 10).

Table 15: Pseudo-code for the procedure CreateNewPopulation

```
;Global constants and variables
INIT_POP_SIZE = 20
MAX_POP_SIZE = INIT_POP_SIZE * 40
MinNumOfSolutions = 0

Procedure CreateNewPopulation(NewPopIndex)
;1) first population
If  NewPopIndex = 0
  ;2) population size is proportional with number of objectives
  If number of objective functions <= 2
    NewPopSize = INIT_POP_SIZE
  Else
    ;3) bigger population size for number of objective function > 2
    NewPopSize = INIT_POP_SIZE * number of objective functions
  End If
  ;4) this is the required minimum number of non-dominated
  ; solutions in the first rank
  MinNumOfSolutions = NewPopSize;
  ;5) number of generation to run when its turn to run
  NumGenRunEachTime = 1
Else ;not first population
  ;6) new population size is twice the previous population size
  NewPopSize = 2 * Pop[NewPopIndex-1].Size
```

```
  ;7) number of generation to run when its turn to run twice that of the prev. pop.
  NumGenRunEachTime = Pop[NewPopIndex-1].NumGenRunEachTime * 2
End If
;allocate memory for new population
Pop[NewPopIndex] = new population[NewPopSize]
Pop[NewPopIndex].NumGenRunEachTime = NumGenRunEachTime

RandomIx = 0
;8) initialize new population check if previous population has good solutions
If NewPopIndex > 0 And Pop[NewPopIndex-1].ConvergentMetric < 1
  ;9) inject solutions, crossover, and mutation values from previous population
  For ix = 0 to Pop[NewPopIndex-1].Size
    Pop[NewPopIndex].Solutions[ix] = Pop[NewPopIndex-1].Solutions[ix]
  End For
  RandomIx = Pop[NewPopIndex-1].Size
End If
;10) fill new population with random solutions, crossover, and mutation values
For ix = RandomIx to Pop[NewPopIndex].Size
    Pop[NewPopIndex].Solutions[ix] = random solutions
End For
11)  set maximum number of generations = 10 * population size
Pop[NewPopIndex].MaxGeneration = NewPopSize * 10
```

- Investigate the convergent metric: The running convergence metric suggested by Deb & Jain (2002) is investigated in this study because the metric can work with an unknown set of Pareto-optimal solutions and it is computed efficiently. The convergence metric for a population P at generation t (denoted as P_t) is calculated in the following steps (Deb & Jain, 2002):

Step 1: Identify the non-dominated set F_t of P_t.

Step 2: From each point i in F_t, compute the smallest normalized Euclidean distance to reference set P^* (P^*= *non-dominated*($\bigcup_{t=0}^{T} F_t$), where T is the number of generations the algorithm has ran) as follows:

$$d_i = \min_{j=1}^{|P^*|} \sqrt{\sum_{k=1}^{m} \left(\frac{f_k(i) - f_k(j)}{f_k^{\max} - f_k^{\min}} \right)^2} \tag{23}$$

Here, m is the number of objective functions; $f_k^{\max}$ and $f_k^{\min}$ are the maximum and the minimum function values of *k-th* objective function in P^*.

Step 3: Calculate the convergence metric by averaging the normalized distance for all points in F_t:

$$C(P_t) = \frac{\sum_{i=1}^{|F_t|} d_i}{|F_t|} \tag{24}$$

The conclusion of this investigation is that the running convergent metric using a population-agglomeration technique suggested by Deb & Jain is unreliable and inefficient for the ANSGA-II. In the population-agglomeration technique, a reference set is defined as the non-dominated set of all combined non-dominated sets from previous generations up to the current generation. The convergent metric for a population is the calculated by finding the smallest normalized Euclidean distance among all distances from each non-dominated solution in the population to each solution in the reference set. In the ANSGA-II, this technique requires extensive memory and computational resources. Multiple reference sets (with their sizes increasing with population sizes and number of generations) must be maintained for multiple populations. As the population size is getting bigger, the calculation of this technique is unacceptable slow. Moreover, the population-agglomeration technique relies merely on the fact that the algorithm being used is able to approximate to the true Pareto-optimal front eventually; otherwise, the reference set becomes useless. This is the reason the convergent metric using the population-agglomeration technique was applied successfully on two three-objective test problems DTLZ2 and DTLZ5 using NSGA-II with good parameter settings (Deb & Jain, 2002). When this technique was applied in the ANSGA-II, it caused the algorithm failing to converge to the true Pareto-optimal fronts on several complex problems such as POL, KUR, ZDT2, ZDT4, and ZDT6. For example, on a problem with several local Pareto-optimal fronts such as ZDT4 and ZDT6, the reference set was wrongly determined because the search was trapped in a local Pareto-optimal front. Hence, the algorithm terminated prematurely as soon as it obtains good distribution in the non-dominated solutions set.

Therefore, the running convergent metric is not used in the ANSGA-II but a simple work around technique is used instead. This technique relies merely on the fact that the underlying NSGA-II used in ANSGA-II is able to approximate to the true Pareto-optimal front eventually. It simply allows the ANSGA-II to run for a while until the number of non-dominated solutions in the first rank at least equal to the required minimum number of solutions (initial population size), and then the algorithm starts to calculate the diversity metric. This work-around technique appears to work effectively. Pseudo-code for the procedure *CalculatePerformanceMetric*, which describes this work-around technique, is listed in Table 16 below. It should also be emphasized that the parameter control techniques used in this study should be applied to MOEAs, which have been verified that they can find diverse non-dominated solution sets with good convergence on benchmark test problems borrowed from the MOEA literature.

Table 16: Pseudo-code for the procedure CalculatePerformanceMetric

```
;Global constants
ACCEPTABLE_DIVERISTY_VAL = 0.80
ACCEPTABLE CONVERGENT_VAL = 0.0055
DIFF_METRIC_VALUE_THRESHOLD = 0.01
NO_IMPROVEMENT_CNT_MAX = 20

Procedure CalculatePerformanceMetric(PopIndex)
ConvergentMetricValue = 1.0
DiversityMetricValue = 1.0
```

```
;1) calculate diversity metric only if enough solutions
If Pop[PopIndex].Rank[0] ≥ MinNumOfSolutions
  ConvergentMetricValue = ACCEPTABLE _CONVERGENT_VAL;
  If Pop[PopIndex].RunCount ≥ 100 * number of objective functions And
    Pop[PopIndex].DiversityMetricValue ≤ ACCEPTABLE _DIVERSITY_VAL)
    ;2) assume that acceptable convergent metric value is achieved
    Pop[PopIndex].ConvergentMetricValue = ACCEPTABLE _CONVERGENT_VAL
  End If
  ;3) calculate diversity metric
  DiversityMetricValue = CalculateDiversityMetric(PopIndex)
  CombinedMetricValues = ConvergentMetricValue + DiversityMetricValue
  ;4) calculate the difference between current metric and previous metric values
  Diff =(Pop[PopIndex].ConvergentMetricValue +
          Pop[PopIndex]. DiversityMetricValue) - CombinedMetricValues
  ;5) is the population close to acceptable convergence and  distribution
  ;see Table 19 for pseudo-code of IsPopulationCloseToConvergence
  If IsPopulationCloseToConvergence(PopIndex))
    ;6) if the difference falls within the threshold or there is no improvement
    ;then increase ConvergedCnt.
    If (Diff ≤ DIFF_METRIC_VALUE_THRESHOLD Or
       Pop[PopIndex].NoImprovementCnt ≥NO_IMPROVEMENT_CNT_MAX
      ;Increase convergent count
      Pop[PopIndex].ConvergedCnt = Pop[PopIndex].ConvergedCnt + 1
      If Pop[PopIndex].NoImprovementCnt ≥NO_IMPROVEMENT_CNT_MAX
        ;7) reset NoImprovementCnt
        Pop[PopIndex].NoImprovementCnt = 0;
      End If
    Else If Pop[PopIndex].ConvergentMetricValue ≤ ConvergentMetricValue And
          (Pop[PopIndex].DiversityMetricValue ≤ DiversityMetricValue Or
           DiversityMetricValue < 0.0)
      ;8) handle the case where the best metric values have been found but
      ;the later metric values somehow cannot be improved
      Increase = Pop[PopIndex].Size/100*2  ;just an arbitrary formula
      If Increase ≤ 1
        ; for population size ≤ 100
        Increase = 1
      End If
      ;9) the increment is proportional with the population size
      Pop[PopIndex].NoImprovementCnt =
                                   Pop[PopIndex].NoImprovementCnt + Increase;
    End If
  End If
End If
;10) ensure that the metric values are monotonically decreased
If ConvergentMetricValue < Pop[PopIndex].ConvergentMetricValue
  Pop[PopIndex].ConvergentMetricValue = ConvergentMetricValue
```

```
End If
If DiversityMetricValue < Pop[PopIndex].DiversityMetricValue
  Pop[PopIndex].DiversityMetricValue = DiversityMetricValue
End If
```

The performance metric values obtained on the final non-dominated solutions using either NSGA-II or ANSGA-II vary with different problems. On easy problems (such as SCH and FON), the algorithms are able to obtain good metric values (i.e. small values). The obtained metric values are larger on difficult problems such as POL and KUR. In the procedure *CalculatePerformanceMetric*, the acceptable diversity value and the acceptable convergent metric value are defined conservatively as 0.8 and 0.005 respectively. It is expected that at the beginning the diversity metric values will be large. The value of diversity metric is zero for the most uniformly and widely spread of non-dominated solution sets. For any other distribution, the value of the diversity metric would be greater than zero. With successive generations, these values will gradually become smaller because increasingly wide spread non-dominated solutions are obtained. It is also expected that near to the end of the run, the change in the calculated metric values will be small because the population has already converged to Pareto-optimal front with diversity of non-dominated solutions. If the difference of the combined metric value falls within a certain threshold (e.g. *DIFF_METRIC_VALUE_THRESHOLD* = 0.01) respectively for successive runs of the population then the obtained non-dominated solutions can be considered identical. When this condition occurs or when there is no improvement in the obtained metric values (the value of *NoImprovementCnt* reaches its maximum value), the value of *ConvergedCnt* is increased by one (step 6). When the value of *ConvergedCnt* reaches its threshold (i.e. *CONVERGED_COUNT_THRESHOLD* = 3), the algorithm can terminate to prevent wasting unnecessary computational resources (see pseudo-code for the main procedure ANSGA-II in Table 14). The increment of *NoImprovementCnt* is handled in step 8 and 9 when the best metric values has been obtained, but in the later runs the metric values cannot be improved. The value of *NoImprovementCnt* is increased proportionally with the population size to ensure that the algorithm does not waste computational resource on a large population that does not have any improvement on later runs of the algorithm. Step 10 in the procedure above ensures that the metric values are monotonically decreased.

- Investigate and implement the diversity metric: The diversity metric that was used in the study of the NSGA-II (Deb, Pratap et al., 2002) to measure the diversity of the final obtained solutions is investigated in this study. The diversity metric value is calculated as follows:

$$\Delta = \frac{d_f + d_l + \sum_{i=1}^{N-1} \left| d_i - \overline{d} \right|}{d_f + d_l + (N-1)\overline{d}} \tag{25}$$

In the original study of NSGA-II, the parameters d_f and d_l are the Euclidean distance between the known Pareto-optimal extreme solutions and the boundary solutions of the obtained non-dominated set. The ANSGA-II is only useful if it can solve problems without knowing their Pareto-optimal solutions in advance; therefore, the parameters d_f and d_l are modified as the Euclidean distance between the extreme boundary solutions (solutions with smallest and largest function values) and the boundary solutions of the obtained non-dominated set because extreme lower bound and extreme upper bound solutions can be easily calculated using variable bounds (see Table 32 and Table 33 in Appendix A). The parameter d_i in equation (25) above is the Euclidean distance between two consecutive solutions of the obtained non-dominated set. The parameter $\overline{d}$ is the average of all Euclidean distance d_i, where $i = 1, 2, \ldots, (N-1)$, assuming that there are N solutions on the first non-dominated front. For the most uniformly and widely distribution of a non-dominated solution set, the value of Δ is zero. For any other distribution, the value of the metric would be greater than zero.

The diversity metric in equation (25) above can only handle two-objective problems. In the ANSGA-II, the diversity metric for problems with more than two objectives is calculated as the average diversity metric value of all diversity metric values calculated on combinations of objective function pairs. For example, the five-objective problem WATER has ten different pairs of objective functions and the diversity metric value Δ is calculated as: $\Delta = [\Delta(f_1, f_2) + \Delta(f_1, f_3) + \Delta(f_1, f_4) + \Delta(f_1, f_5) + \Delta(f_2, f_3) + \Delta(f_2, f_4) + \Delta(f_2, f_5) + \Delta(f_3, f_4) + \Delta(f_3, f_5) + \Delta(f_4, f_5)] / 10)$. Pseudo-code for the procedure *CalculateDiversityMetric* is presented in Table 17.

Table 17: Pseudo-code for the procedure CalculateDiversityMetric

```
Procedure CalculateDiversityMetric(PopIndex)
;initialize variables
DiversityMetricValue = 0
CountPairs = 0
For f1= 1 to (number of objective functions –1)
  N = 0       ;initialize N to zero for each pair of objective functions
  For  ix=0 to Pop[PopIndex].Size
    ; take only solutions in the first non-dominated front and have no constraint errors
    f1_fitness[N] = Pop[PopIndex].Solutions[ix].fitness[f1];
    sortedIndexes[N] = ix;
    N = N + 1     ;keep track number of valid solutions
  End For

  ;sort fitness values of the first function in ascending order
  ;the sorted indexes are stored in sortedIndexes in descending order
  ;for retrieving fitness values of the second function
  quickSort (f1_fitness, sortedIndexes, 0, N-1);

  For f2 = f1+1 to number of objective functions
    For ix=0 to Pop[PopIndex].Size
      sortedIx = sortedIndexes[ix];
```

```
      f2_fitness[ix] = Pop[PopIndex].Solutions[sortedIx].fitness[f2];
    End For
    ;calculate diversity metric value on two array of fitnesses: f1_fitness and f2_fitness
    LastIx = Pop[PopIndex].Size - 1
    ;1) calculate d_f assuming fExtreme was calculated at the start of the program
    df = sqrt(pow((fExtreme[f1].min - f1_fitness[0])/ fExtreme[f1].min, 2) +
              pow((fExtreme[f2].max - f2_fitness[0])/ fExtreme[f2].max, 2));
    ;2) calculate d_l
    dl = sqrt(pow((fExtreme[f1].max - f1_fitness[LastIx])/ fExtreme[f1].max, 2) +
              pow((fExtreme[f2].min - f2_fitness[LastIx])/ fExtreme[f2].min, 2));
    ;3) calculate all Euclidean distances
    Skip = 0    ;keep track of number of zero distances (duplicate solutions) or gaps
    For ix=0 to  N - 1
      Dist = sqrt(pow((f1_fitness[ix] - f1_fitness[ix+1]), 2) +
                  pow((f2_fitness[ix] - f2_fitness[ix+1]), 2));
      ;4) with duplicate non-dominated solutions Dist = 0 and
      ;for the problems with discontinuous Pareto-optimal fronts (e.g. POL, KUR,
      ;ZDT3, skip the gap on the Pareto-optimal front in order to calculate the
      ;diversity metric value correctly
      If  Dist = 0 Or (TotalDistance > 0 And
                  Dist ≥ 1000*distances[ix-Skip-1])) ;expect a big gap.
        Skip = Skip + 1    ;keep track of skip count
      Else
        distances[ix-Skip] = Dist
        TotalDistance = TotalDistance + distances[ix-Skip]
      End If
      N = N - Skip       ;actual number of valid and non-duplicate solutions
      ;5) calculate average of all distances d̄
      d̄ = TotalDistance / (N-1);
      ;calculate Σ_{i=1}^{N-1} |d_i - d̄|
      sum = 0;
      For ix=0 to N-1
        sum = sum+ |distances[ix] – d̄ |
      End For
      ;6) calculate diversity metric value for each objective function pair as in eq. (25)
      MetricValuePerPair = (df + dl + sum) / (df + dl + ((N-1)*d̄ ));
      DiversityMetricValue = DiversityMetricValue + MetricValuePerPair
   End For     ;loop for f_2
   CountPairs = CountPairs + 1
End For        ;loop for f_1
;7) if number of objectives > 2, calculate diversity metric value as average of
;diversity metric values for all objective function pairs
If number of objective functions > 2
  DiversityMetricValue = DiversityMetricValue / (CountPairs)
End If
```

In the step 4 of the procedure *CalculateDiversityMetric*, duplicated solutions and gaps between discontinuous Pareto-optimal fronts are removed from the diversity metric calculation in order to achieve accurate calculations. Duplicated solutions are recognized when the Euclidean distance between two consecutive solutions is zero. A big gap between two discontinuous Pareto-optimal fronts is recognized when the Euclidean distance between two consecutive solutions is 1000 times greater than the previous distance. The value of 1000 is chosen because after several experiments it is found to be a good value for determining a gap. However, the author of this dissertation feel that a better way should be investigated to distinguish between valid gaps in problems with discontinuous Pareto-optimal fronts and invalid gaps between two non-dominated solutions in problems with continuous Pareto-optimal fronts. The modified diversity metric appears to work effectively in comparing two or more non-dominated solution sets during the execution of the ANSGA-II and it enables the ANSGA-II to select proper population sizes for the problems being solved.

- Implement the better solutions condition: A solution set *A* is better than a solution set *B* if the following condition is satisfied:

$$C_A(P_t) + \Delta_A < C_B(P_t) + \Delta_B \tag{26}$$

In the above equation, $C_A(P_t)$ and $C_B(P_t)$ are convergence metrics for two solution sets *A* and *B* respectively; Δ_A and Δ_B are diversity metrics for solution set *A* and *B* respectively. Note that the values for Δ_A and Δ_B are not actually calculated but they are assigned good convergent metric values after the ANSGA-II has run for a while and when the number of non-dominated solutions in the first rank at least equal to the required minimum number of solutions (see the procedure *CalculatePerformanceMetric* in Table 16). Pseudo-code for the procedure *CompareTwoSolutionSets* is presented in Table 18. This procedure simply returns the index of the population, which has a better non-dominated solution set.

Table 18: Pseudo-code for the procedure CompareTwoSolutionSets

```
Procedure CompareTwoSolutionSets (PopIndex1, PopIndex2)
Pop1CombinedMetricValue = Pop[PopIndex1].ConvergentMetricValue +
                          Pop[PopIndex1].DiversityMetricValue
Pop2CombinedMetricValue = Pop[PopIndex2].ConvergentMetricValue +
                          Pop[PopIndex2].DiversityMetricValue
If Pop1CombinedMetricValue ≤ Pop2CombinedMetricValue And
                  Pop[PopIndex1].Rank[0] ≥ MinNumOfSolutions
  return PopIndex1        ;population 1 has better solutions
Else If Pop2CombinedMetricValue ≤ Pop1CombinedMetricValue And
                       Pop[PopIndex2].Rank[0] ≥ MinNumOfSolutions
  return PopIndex2        ;population 2 has better solutions
End If
```

- Implement the close to convergence condition: In the ANSGA-II context, the close to convergence condition implies that the population has number of non-dominated solutions in the first non-dominated rank equal or greater than the required minimum number of solutions (initial population size) and its convergent count is greater than zero (the population is converged when its convergent count reached maximum value = 3), or both the convergent and diversity metric values are less than or equal to the acceptable values. Pseudo-code for the procedure *IsPopulationCloseToConvergence* is presented in Table 19.

Table 19: Pseudo-code for the procedure IsPopulationCloseToConvergence

```
Procedure IsPopulationCloseToConvergence(PopIndex)
  GoodEnough = Pop[PopIndex].ConvergedCnt > 0) Or
                  (Pop[PopIndex].DiversityMetricValue ≤
                              ACCEPTABLE_DIVERSITY_METRIC_VALUE And
                   Pop[PopIndex].ConvergentMetricValue ≤
                              ACCEPTABLE CONVERGENTMETRIC_VALUE
```

Return GoodEnough

- Incorporate the self-adaptive crossover parameter into the ANSGA-II: The procedure SBX listed in Table 11 is modified. The crossover probability (p_c) and distribution index (η_c) for the SBX operator are attached to each solution in the population and allowed to co-evolve with each solution. The crossover parameter values and performance metric values are observed to make sure that the adaptive crossover parameter values improve the algorithm performance instead of destroying good non-dominated solutions. The diversity metric values are expected to decrease monotonically because the smaller diversity metric value is the better distribution in the obtained solutions. Pseudo-code for the modified procedure SBX is listed in Table 20.

Table 20: Pseudo-code for the modified procedure SBX

```
Procedure SBX (M_t, Q_t)
p_c = 0.0        ;crossover probability
η_c = 0.0        ;distribution index for real-coded crossover
n = 0
Repeat
   p_{c_n} = M_{t_n}^{p_c}        ;retrieve crossover probability attached to n-th parent from M_t
   η_{c_n} = M_{t_n}^{η_c}        ;crossover distribution index attached to n-th parent from M_t
   p_{c_{n+1}} = M_{t_{n+1}}^{p_c}  ;crossover probability attached to (n+1)-th parent from M_t
   η_{c_{n+1}} = M_{t_{n+1}}^{η_c}  ;crossover distribution index attached to (n+1)-th parent from M_t
   ;use crossover values from the n-th parent in M_t by default
   p_c = p_{c_n}
   η_c = η_{c_n}
   random = random number ∈ [0,1]
   ;should crossover values from the (n+1)-th parent in M_t be used instead?
   If (random ≤ p_{c_{n+1}} and random > p_{c_n}) or
     (random ≤ p_{c_n} and random ≤ p_{c_{n+1}} and p_{c_{n+1}} > p_{c_n})
     p_c = p_{c_{n+1}}
     η_c = η_{c_{n+1}}
   End If
   If random ≤ p_c               ;decide to do crossover or not
```

```
;four more variables p_c, η_c, p_m, and η_m are attached to the end of each
;solution M_{t_n} and subject to crossover with decision variables
  For each decision variable and parameter variable i ∈ M_{t_n} Do
    ;select two parents from mating pool
    x_i^(1,t) = M_{t_n}^i        ;i-th variable of the n-th parent from M_t
    x_i^(2,t) = M_{t_{n+1}}^i    ;i-th variable of the (n+1)-th parent from M_t
    r_i = random number ∈ [0,1]
    If r_i ≤ 0.5
      β_{q_i} = (2r_i)^(1/(η_c+1))
    Else
      β_{q_i} = (1/(2(1-r_i)))^(1/(η_c+1))
    End If
    ;perform crossover on two parents (x_i^(1,t) and x_i^(2,t)) then store
    ;two offspring solutions in new population Q_t
    ;Q_{t_n}^i is i-th variable of the n-th solution in Q_t
    ;Q_{t_{n+1}}^i is i-th variable of the (n+1)-th solution in Q_t
    Q_{t_n}^i = 0.5[(1+β_{q_i}) x_i^(1,t) + (1-β_{q_i}) x_i^(2,t)]
    Q_{t_{n+1}}^i = 0.5[(1-β_{q_i}) x_i^(1,t) + (1+β_{q_i}) x_i^(2,t)]
  End For
Else    ;do not do crossover
  For each decision variable and parameter variable i ∈ M_{t_n} Do
    ;copy solutions from mating pool to new population
    Q_{t_n}^i = M_{t_n}^i            ;Q_{t_n}^i is i-th variable of the n-th solution in Q_t
    Q_{t_{n+1}}^i = M_{t_{n+1}}^i    ;Q_{t_{n+1}}^i is i-th variable of the (n+1)-th solution in Q_t
  End For
End If
n = n + 2
Until n ≥ (population size N)/2
Return Q_t
```

- Incorporate the self-adaptive mutation parameter into the ANSGA-II: The Polynomial Mutation procedure listed in Table 12 is modified. The mutation probability (p_m) and distribution index (η_m) for the polynomial mutation operator are attached to each solution in the population and allowed to co-evolve with each solution. The mutation parameter values and diversity metric values are observed to make sure that the adaptive values improve the algorithm performance instead of destroying good non-dominated solutions. The diversity metric values are expected to decrease monotonically because the smaller diversity metric value is the better distribution in the obtained solutions. Pseudo-code for the modified *PolynomialMutation* procedure is listed in Table 21.

Table 21: Pseudo-code for the modified procedure PolynomialMutation

```
Procedure PolynomialMutation (Q_t)
p_m = 0.0      ;mutation probability
η_m = 0.0      ;distribution index for real-coded mutation
n = 0
Repeat
  p_m = M_{t_n}^{p_m}     ;mutation probability is attached to each solution
  η_m = M_{t_n}^{η_m}     ;distribution index for mutation is attached to each solution
  ;four more variables p_c, η_c, p_m, and η_m are attached to the end of each
  ;solution Q_{t_n} and subject to mutation with decision variables
  For each decision variable and parameter variable x_i ∈ Q_{t_n}  Do
    r_i = random number ∈ [0,1]
    If r_i ≤ p_m              ;decide to do mutation or not
      If x_i > x_i^(L)        ; x_i^(L) is lower bound of x_i
        r_i = random number ∈ [0,1]
        If r_i ≤ 0.5
          δ̄_i = (2r_i)^(1/(η_m+1)) − 1
        Else
          δ̄_i = 1 − [2(1 − r_i)]^(1/(η_m+1))
        End If
        ; Q_{t_n}^i is i-th variable of the n-th solution in Q_t
        Q_{t_n}^i = x_i + (x_i^(U) − x_i^(L)) × δ̄_i    ;mutate i-th variable of x
      Else     ; x_i ≤ x_i^(L)
        r_i = random number ∈ [0,1]
        ; Q_{t_n}^i is i-th variable of the n-th solution in Q_t
        Q_{t_n}^i = r_i × (x_i^(U) − x_i^(L)) + x_i^(L)    ;mutate i-th variable of x
      End If
```

```
    End If
  End For
  n = n + 1
Until n ≥ population size N
Return Q_t
```

- Evaluate the ANSGA-II: The ANSGA-II is executed to solve each test problem one at a time. Plotting graphs of the obtained non-dominated solutions on each test problem together with number of generation taken, diversity metric values for the obtained solutions on each test problem are used to evaluate the ANSGA-II against the NSGA-II based on their ability to converge to the Pareto-optimal front and diversity of the obtained non-dominated solutions. The format for presenting the results is described in the next section.

Format for Presenting Results

Results are presented in the forms of graphs and tables to show the performance of the ANSGA-II against the original NSGA-II. This section presents performance results of the ANSGA-II against the NSGA-II in the table format. It then describes the plotting presentations of the obtained non-dominated solutions for two-objective test problems and for the five-objective real-world problem named WATER.

Presentation of Performance Results

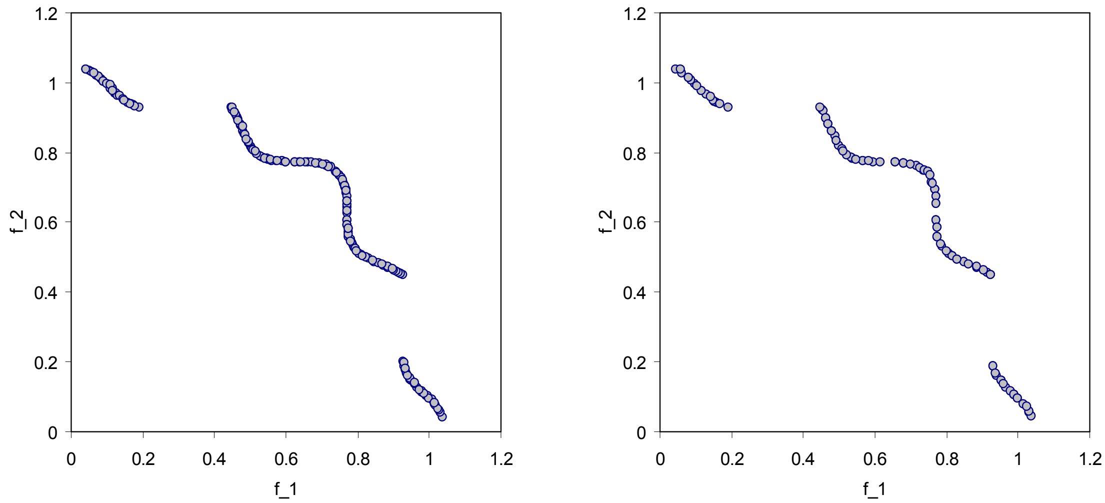
Table 22 below provides a sample of how the performance results of the ANSGA-II and NSGA-II are presented and compared on thirteen benchmark multi-objective problems used in this study.

Table 22: Sample of performance results of ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) against the original NSGA-II with fixed parameter settings

Problems	ANSGA-II						NSGA-II					
	N	G	Diversity Metric	Func. Eval.	Time (sec)	N*G (1000)	N	G	Diversity Metric	Func. Eval.	Time (sec)	N*G (1000)
SCH	80	300	0.5503	591	17	24	100	250	0.5711	251	4	25
FON	80	232	0.7203	663	16	18.56	100	250	0.7219	251	4	25
POL	160	456	0.9518	1215	76	72.96	100	250	0.9538	251	3	25
KUR	320	352	0.8005	1200	162	112.64	100	250	0.8121	251	4	25
ZDT1	40	280	0.6760	608	19	11.2	100	250	0.7431	251	4	25
ZDT2	40	346	0.7010	678	12	13.84	100	250	0.7390	251	6	25
ZDT3	40	286	0.7118	578	12	11.44	100	250	0.8731	251	5	25
ZDT4	160	352	0.7426	899	71	56.32	100	250	0.6766	251	5	25
ZDT6	40	364	0.6751	748	20	14.56	100	250	0.7046	251	8	25
DEB	80	344	0.7065	886	37	27.52	100	500	0.7772	501	15	50
SRN	40	248	0.7380	488	9	9.92	100	500	0.8011	501	16	50
TNK	160	552	0.8095	1391	80	88.32	100	500	0.8072	501	8	50
WATER	400	540	0.6597	825	340	216	100	500	0.6274	501	10	50

Plots for Two-Objective Problems

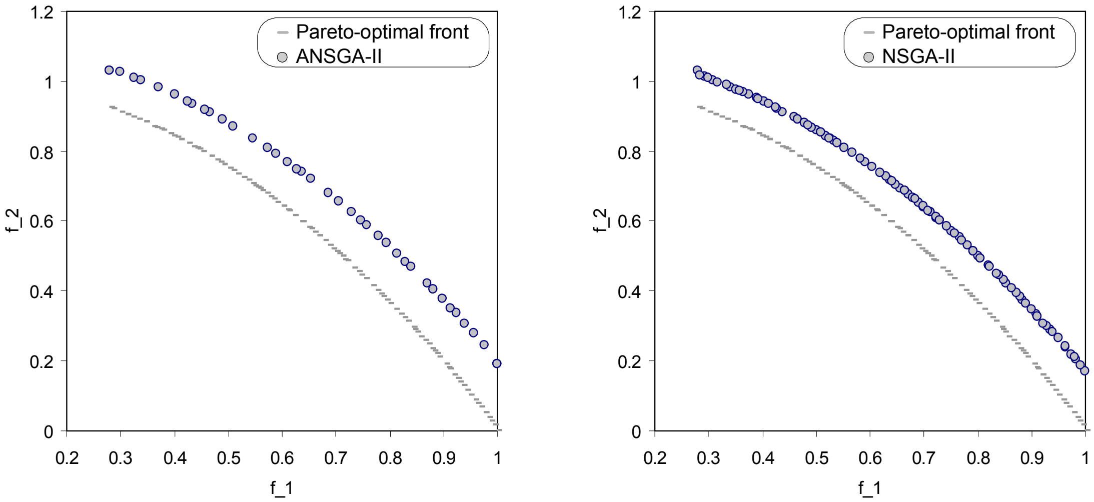
The obtained non-dominated solutions in objective space on two-objective test problems are plotted with two axes represent two objective-functions f_1 and f_2 respectively. Figure 9(a) and Figure 9(b) provide a sample that shows the obtained non-dominated solutions in objective space on the two-objective constrained test problem TNK (see Appendix A) obtained by the ANSGA-II with adaptable parameters (N, p_c, p_m, η_c, η_m) and NSGA-II with fixed parameter settings ($N = 100$, $G = 500$, $p_c = 0.9$, $p_m = 0.5$, $\eta_c = 20$, $\eta_m = 100$) respectively. The figures visually present each MOEA's overall qualitative performance and their results can be compared visually.



(a) (b)

Figure 9: Sample of plotting of obtained non-dominated solutions on the two-objective test problem TNK with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

For the problem TNK, both ANSGA-II with adaptable parameters and NSGA-II with fixed parameter settings are able to converge to the true Pareto-optimal front; therefore, the true Pareto-optimal front is not explicitly shown in Figure 9(a) and Figure 9(b). If neither algorithm fails to converge to the true Pareto-optimal front as in the case for the test problem ZDT6, the true Pareto-optimal front is explicitly plotted as shown in Figure 10 below.



(a) (b)

Figure 10: Sample of plotting of non-dominated solutions on the test problem ZDT6 with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Plots for the Five-Objective Problem WATER

It is difficult to visualize a graph with more than 3-axes. Therefore, for the five-objective real-world problem WATER (see Appendix A), the scatter-plot matrix method as suggested by Meisel (1993) and Cleveland (1994) is used to plot the obtained non-dominated solutions. This method plots all $\begin{pmatrix} m \\ 2 \end{pmatrix}$ pairs of plots among m objective functions. In order to compare the obtained solutions among the five-objective functions in the test problem WATER, there are total of 5 * 2 or 10 plots for both algorithms ANSGA-II and NSGA-II as shown in Figure 11. In Figure 11, the label and ranges used for each axis are shown in the diagonal boxes. Upper diagonal plots are for ANSGA-II and lower diagonal plots are for NSGA-II. Thus, a (i, j) plot for ANSGA-II with $i < j$ can be compared with a (j, i) plot for NSGA-II. For example, the sub-plot in position (3, 4) for ANSGA-II has its vertical axis marked as f_3 with range 0.0 to 1.0 and the horizontal axis marked as f_4 with range 0.0 to 1.6 can be compared with the sub-plot in position (4,

3) for NSGA-II, which has its horizontal axis marked as f_3 with range 0.0 to 1.0 and the vertical axis marked as f_4 with range 0.0 to 1.6.

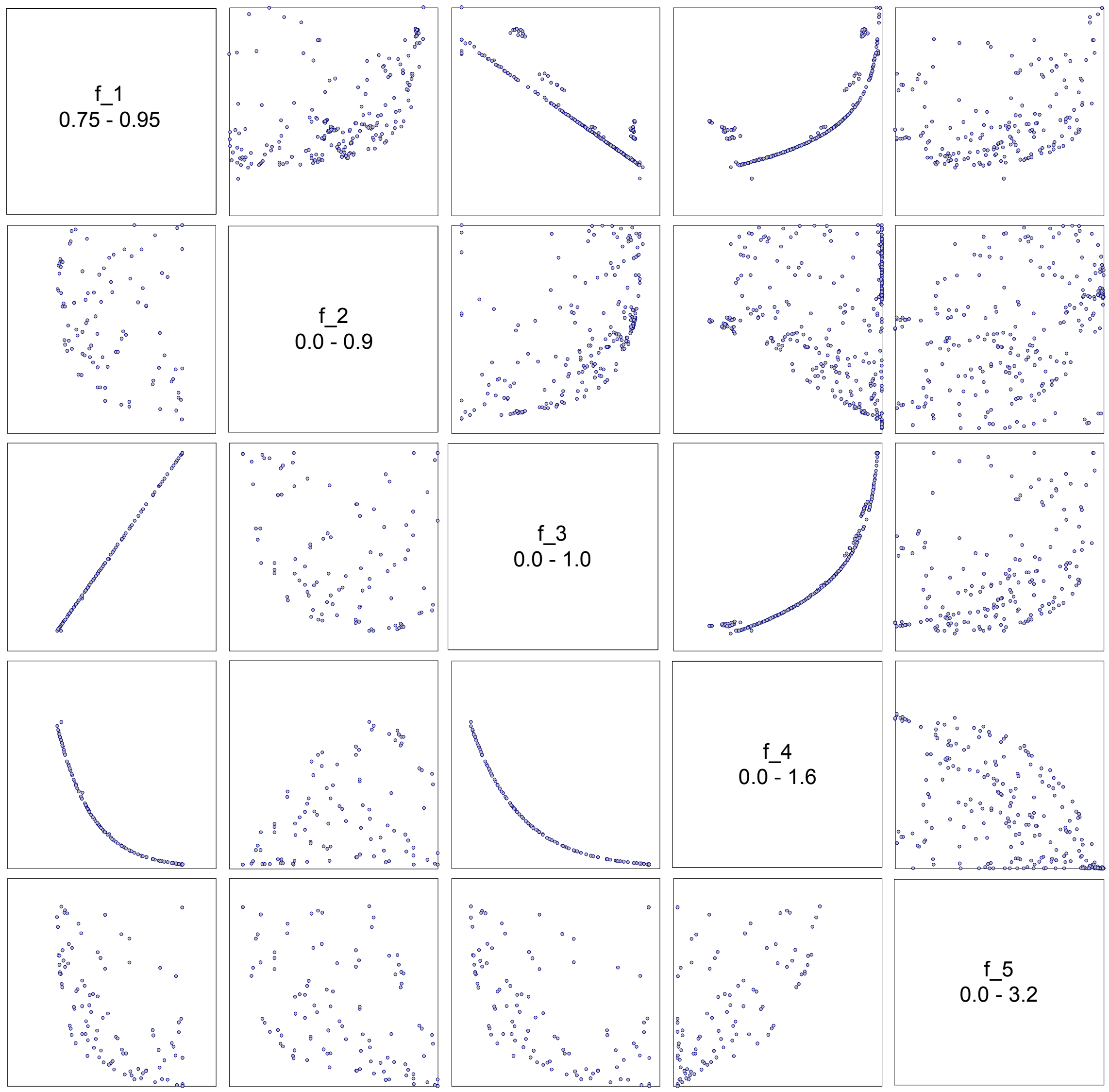


Figure 11: Sample of plotting of non-dominated solutions on the five-objective real-world problem WATER with upper diagonal plots for ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and lower diagonal plots for NSGA-II with fixed parameter settings

Resource Requirements

The following is a list of resources required to complete this dissertation:

- A Pentium class PC with at least 1.59 GHz and 512 MB of RAM.

- Microsoft Visual C++ 6.0 or Microsoft Visual C++ .NET is used to develop the algorithms in this dissertation.
- Microsoft Visual SourceSafe 6.0 is used to control source code and back up documents.
- Microsoft Word 2000 is used to prepare the documents.
- Microsoft Excel 2000 is used to plot the graphs.
- Microsoft Visio Professional 2000 is used to prepare figures in the documents.
- EndNote 6.0 is used to manage references and format citations.
- MathType 5.2c or above is used to prepare equations in the documents.

Reliability and Validity

The ANSGA-II is evaluated using the same suite of benchmark multi-objective test problems that were used in the study of the original NSGA-II. Since these test problems are selected from a number of significant past studies in MOEA (Deb, Pratap et al., 2002) and most of Pareto-optimal solutions are known for these test problem (except for the problem WATER), the non-dominated solutions obtained by the ANSGA-II on these test problems can be validated reliably. The diversity metric used in the study of the NSGA-II is modified to handle un-known Pareto-optimal sets and problems with more than two objectives. For compatible evaluation of the ANSGA-II against the NSGA-II, this modified diversity metric is also used to calculate diversity metric values of the final non-dominated solution sets obtained by the NSGA-II on the test problems used in this study. Therefore, the diversity metric values for the NSGA-II presented in this dissertation cannot be validated reliably with those published in the study of the NSGA-II (Deb, Pratap et al., 2002). The implementation of the diversity metric used in this study, however, may have some unavoidable defects. Since the source code for this metric is not available, the implemented code for this metric is based on its descriptions from the literature (Deb & Jain, 2002; Deb, Pratap et al., 2002). Variations from the original source code due to misinterpretation of the authors' original intent, unintentional omissions, and gaps in the descriptions of the metric are possible. Every effort has been attempted to provide the most valid and reliable results possible.

Summary

To summarize this chapter, it should be emphasized that the study uses evaluation research method, which consists of the following main steps. Formative studies of existing parameter control techniques and the NSGA-II are performed to identify the available techniques that can be used, issues and barriers that are needed to be resolved. The new algorithm ANSGA-II is then developed based on the NSGA-II. The ANSGA-II is evaluated against the original NSGA-II using the same benchmark multi-objective test problems that were used in the study of the NSGA-II. Since the same test problems are used, the results generated by the ANSGA-II can be easily compared to that of the NSGA-II for validation. This chapter continues by presenting the specific procedures that are used to conduct the study. First, the original NSGA-II is presented in details. Then, the changes to the NSGA-II to make the ANSGA-II are explained. Results are presented in the forms of graphs and tables to show the performance of the ANSGA-II against the original NSGA-II. The plotting presentations of the obtained non-dominated solutions for two-objective test problems and for the five-objective real-world problem named

WATER are discussed. Samples of the plots and table presentations of performance results are also presented. This chapter then lists the resource requirements for the study. Finally, a discussion of reliability and validity of the results is provided.

Chapter 4

Results

This chapter presents the comparison of performance results of the ANSGA-II with three adaptable parameters (population size, crossover probability, and mutation probability) against the original NSGA-II on thirteen benchmark multi-objective problems in terms of diversity metric, number of function evaluations, execution time, number of solutions evaluated, and quality of the Pareto-optimal front. The performance measures are described in the following.

- Diversity metric: The diversity metric values that are calculated on the final non-dominated solutions obtained by the ANSGA-II and the NSGA-II respectively. A small diversity metric value indicates better distribution of non-dominated solution set.
- Number of function evaluations: For the ANSGA-II, this value represents the total number of function evaluations that the ANSGA-II executed to find the non-dominated solutions and proper values for adaptable parameters. For the NSGA-II, this value represents the total number of function evaluations that the NSGA-II executed to find the non-dominated solutions. The number of function evaluations for the ANSGA-II is always equal or greater than that of the NSGA-II due to overheads of executing multiple populations to find a proper population size for the problem being solved.
- Execution time: For the ANSGA-II, this value is the total time in seconds that the ANSGA-II executed to find the non-dominated solutions and proper values

for adaptable parameters. For the NSGA-II, this value is the total time in seconds that the NSGA-II executed to find the non-dominated solutions. In most problems, the ANSGA-II has longer execution time than the NSGA-II due to overheads of solving the problem and learning good parameter values at the same time. In a small number of problems (e.g. SRN and DEB as presented later), the ANSGA-II has shorter execution time than the NSGA-II indicating that the overhead for learning good parameter values for these problems is minimum.

- Number of solutions evaluated ($N*G$): N is the population size and G is the number of generations executed on the population. This value represents a measure of the actual number of solutions evaluated by the algorithm to find the non-dominated solutions (excluding number of solutions evaluated for finding a proper population size). Hence, a small value is better. In most problems, when the ANSGA-II has a smaller population size, its associated number of solutions evaluated is also smaller than that of the NSGA-II. In a few problems, the ANSGA-II has a smaller population size but the number of generations is higher. Consequently, the ANSGA-II has its number of solutions evaluated higher than that of the NSGA-II.
- Quality of the Pareto-optimal front: is determined by using a combination of visual observation of plots and diversity metric values. Based on the observations on several runs of the ANSGA-II in this study, the following terms are defined for the sake of discussion:

Adequate spread or adequate distribution or adequate diversity metric value: when the diversity metric value ≤ 0.8 (as implemented in the ANSGA-II).

Better spread or better distribution or better diversity metric value: A diversity metric value dm_1 is better than dm_2 if $dm_1 \leq dm_2 - 0.001$.

Equivalent spread or equivalent distribution or equivalent diversity metric value: A diversity metric value dm_1 is equivalent to a diversity metric value dm_2 if ($dm_1 > dm_2$ and $dm_1 \leq dm_2 + 0.005$) or ($dm_1 \leq dm_2$ and $dm_1 > dm_2 - 0.001$).

Equivalent convergence: The equivalent convergence is compared visually between two plots of non-dominated solutions in objective space: one is obtained by the ANSGA-II and another one is obtained by the NSGA-II for the problem being solved.

Satisfactory convergence or adequate convergence: is the case when one or two non-dominated solutions do not line up on the Pareto-optimal front.

Inferior convergence: Convergence is not close to the true Pareto-optimal front or several non-dominated solutions do not line up on the Pareto-optimal front.

In addition to running the ANSGA-II with three adaptable parameters (population size, crossover probability, and mutation probability), this dissertation ran experiments where only subsets of these three parameters are adaptable for studying which adaptable parameter affects the performance of the ANSGA-II the most or the least (i.e. whether the execution time is reduced if fewer parameters are adaptable or which adaptable parameter affects the quality of the Pareto-optimal front). Table 23 presents seven variations of the ANSGA-II. Since, real-coded crossover operator (SBX) and real-coded mutation operator (polynomial mutation operator) are used in the ANSGA-II, adaptable p_c implies that its associated distribution index η_c is adaptable as well. Likewise, adaptable p_m implies that its associated distribution index η_m is also adaptable.

Table 23: Seven variations of ANSGA-II

Variation	Population Size	Crossover Probability	Mutation Probability
0	Adaptable	Adaptable	Adaptable
1	Adaptable	Adaptable	
2	Adaptable		Adaptable
3		Adaptable	Adaptable
4	Adaptable		
5		Adaptable	
6			Adaptable

The performance results of these variants are presented next. The chapter ends with a summary of results.

Findings

First, this section provides a summary of performance comparisons of the ANSGA-II and its six variants against the NSGA-II with fixed parameter settings as shown in Table 24. It then provides detail presentations of the performance results.

Table 24: Summary of performance comparison of ANSGA-II and six variants of ANSGA-II against the NSGA-II using the original parameter settings

Problems	ANSGA-II with adaptable						
	N, p_c, p_m	N, p_c	N, p_m	p_c, p_m	N	p_c	p_m
SCH	**better**	adequate	**better**	**better**	adequate	**better**	**better**
FON	**better**	adequate	adequate	adequate	**better**	**better**	**better**
POL	**better**	*worse*	**better**	**better**	**better**	same	same
KUR	**better**	**better**	*worse*	*worse*	**better**	*worse*	*worse*
ZDT1	**better**	adequate	**better**	**better**	adequate	**better**	**better**
ZDT2	**better**	**better**	*worse*	**better**	**better**	adequate	same
ZDT3	**better**	adequate	**better**	**better**	*worse*	same	**better**
ZDT4	**better**	*worse*	*worse*	*worse*	*worse*	**better**	**better**
ZDT6	**better**	same	*worse*	*worse*	*worse*	same	*worse*

DEB	**better**	*worse*	adequate	**better**	adequate	adequate	**better**
SRN	adequate	adequate	**better**	**better**	**better**	**better**	**better**
TNK	**better**	**better**	worse	adequate	worse	**better**	adequate
WATER	**better**	worse	**better**	worse	worse	worse	adequate

A combination of visual observation of plots (presented later) and performance measures (described above) is used for performance comparisons. The entries in the table are described in the following.

- **better**: (i) better convergence and better or equivalent diversity with reasonable execution time; or (ii) equivalent convergence, better diversity, and better or reasonable execution time compared to that of required by the NSGA-II.
- same: equivalent convergence and equivalent diversity with a smaller population size or the same population size.
- adequate: satisfactory convergence (one or two non-dominated solutions may not line up on the Pareto-optimal front) and better or equivalent diversity with a smaller population size.
- *worse:* (i) inferior convergence and/or inferior diversity of non-dominated solutions; or (ii) better convergence and/or better diversity but it takes unacceptable long time to finish.

Table 24 shows that the ANSGA-II with adaptable N, p_c, and p_m is the winner. It out-performs the original NSGA-II with fixed parameter settings and six other variants. The other six variants of the ANSGA-II perform worse than the original NSGA-II. Based on the comparison in Table 24, the following ranks (1 for being the best) can be classified on the algorithms in terms of finding diverse non-dominated solution set and converging to Pareto-optimal front: (1) ANSGA-II with all three adaptable parameters N, p_c, p_m; (2) NSGA-II using original parameter settings; (3) ANSGA-II with adaptable mutation probability alone; (4) ANSGA-II with adaptable crossover probability alone; (5) ANSGA-II with adaptable crossover probability and mutation probability; (6) ANSGA-II with adaptable population size and crossover probability; (7) ANSGA-II with adaptable population size and mutation probability; (8) and ANSGA-II with adaptable population size only. Regarding the overhead for adapting parameters and solving the problem at the same time (presented later), the variants with adaptable population size N (i.e. variants 0, 1, 2, and 4) take longer time than other variants (i.e. variants 3, 5, and 6) due to overhead of executing multiple populations simultaneously for learning a proper population size. Other variants without adaptable population size (i.e. variants 3, 5, and 6) have execution time comparable to that of required by the NSGA-II. This implies that the ANSGA-II is able to learn good values for crossover probability and mutation probability quickly.

In the following detail presentations, the fixed parameter settings for the NSGA-II are set to the same values that were used in the study of the original NSGA-II (Deb, Pratap et al., 2002): $N = 100$; $p_c = 0.9$; $p_m = 0.5$; $\eta_c = 20$; $\eta_m = 100$; $G = 250$ for problems SCH, FON, POL, KUR, ZDT1, ZDT2, ZDT3, ZDT4; $G = 500$ for problems ZDT6 DEB, SRN, TNK, WATER. For each variant of the ANSGA-II, performance results on thirteen multi-objective problems used in this study (see Table 32 and Table 33 in Appendix A) and plots of the obtained non-dominated solutions are presented and compared against those of the original NSGA-II. A brief description of each problem is provided in the first sub-section "Results of ANSGA-II with Adaptable N, p_c, p_m, η_c, η_m."

Results of ANSGA-II with Adaptable N, p_c, p_m, η_c, η_m

This is a full version of ANSGA-II, which supports adaptive population size (N), self-adaptive crossover probability (p_c), and self-adaptive mutation probability (p_m). The crossover distribution index (η_c) and mutation distribution index (η_m), which support real-coded crossover operator (SBX) and real-coded mutation operator (polynomial mutation), are also self-adaptive. Table 25 presents performance results of the ANSGA-II with adaptable N, p_c, p_m, η_c, η_m against the original NSGA-II with fixed parameter settings on thirteen benchmark problems.

Table 25: Performance results of ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) against the original NSGA-II with fixed parameter settings

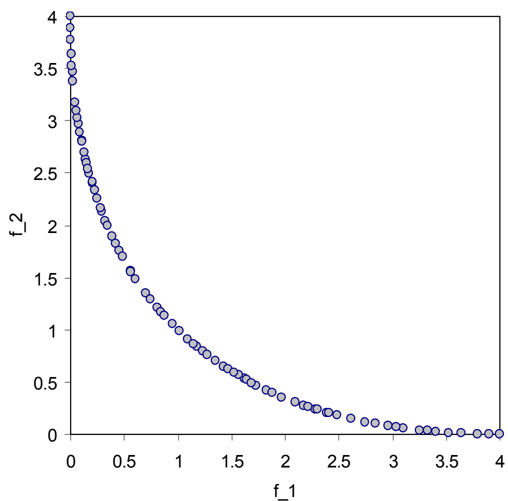
Problems	ANSGA-II						NSGA-II					
	N	G	Diversity Metric	Func. Eval.	Time (sec)	$N*G$ (1000)	N	G	Diversity Metric	Func. Eval.	Time (sec)	$N*G$ (1000)
SCH	80	300	0.5503	591	17	24	100	250	0.5711	251	4	25
FON	80	232	0.7203	663	16	18.56	100	250	0.7219	251	4	25
POL	160	456	0.9518	1215	76	72.96	100	250	0.9538	251	3	25
KUR	320	352	0.8005	1200	162	112.64	100	250	0.8121	251	4	25
ZDT1	40	280	0.6760	608	19	11.2	100	250	0.7431	251	4	25
ZDT2	40	346	0.7010	678	12	13.84	100	250	0.7390	251	6	25
ZDT3	40	286	0.7118	578	12	11.44	100	250	0.8731	251	5	25
ZDT4	160	352	0.7426	899	71	56.32	100	250	0.6766	251	5	25
ZDT6	40	364	0.6751	748	20	14.56	100	250	0.7046	251	8	25
DEB	80	344	0.7065	886	37	27.52	100	500	0.7772	501	15	50
SRN	40	248	0.7380	488	9	9.92	100	500	0.8011	501	16	50
TNK	160	552	0.8095	1391	80	88.32	100	500	0.8072	501	8	50
WATER	400	540	0.6597	825	340	216	100	500	0.6274	501	10	50

On all thirteen benchmark multi-objective problems, the ANSGA-II with adaptable N, p_c, p_m, η_c, η_m out-performs the original NSGA-II with fixed parameter settings in term of finding a diverse set of non-dominated solutions. In term of converging to the true Pareto-optimal front, the ANSGA-II with adaptable N, p_c, p_m, η_c, η_m performs better or as well as the original NSGA-II. For the problem ZDT4, the ANSGA-II is able to converge to the global Pareto-optimal front while the NSGA-II converges to a local Pareto-optimal front. Both ANSGA-II and NSGA-II fail to converge to the global Pareto-optimal front in solving the problem ZDT6. For all other problems, both ANSGA-II and NSGA-II are able to converge to the true Pareto-optimal front except for the problem WATER, which the true Pareto-optimal front is unknown. The ANSGA-II is able to find proper parameter values for N, p_c, p_m, η_c, η_m and adjust these values during its run in solving the problems effectively. For examples, the ANSGA-II requires smaller population sizes (less number of solutions evaluated) than the NSGA-II to solve the simple problems (i.e. SCH, FON, ZDT1, ZDT2, ZDT3, DEB, and SRN) and larger population sizes (more number of solutions evaluated) to solve complex problems (i.e. POL, KUR, ZDT4, TNK, WATER). The ANSGA-II requires less time to solve the problem SRN than the NSGA-II. For other problems, the ANSGA-II is slower than the NSGA-II due to overheads, which appear to be acceptable, of solving the problem and learning good parameter values at the same time. However, it takes trial and error experiments to obtain the proper parameter settings for the NSGA-II in solving an arbitrary problem. The plots of non-dominated solutions on thirteen benchmark problems obtained by the ANSGA-II and the NSGA-II are presented and compared in the following.

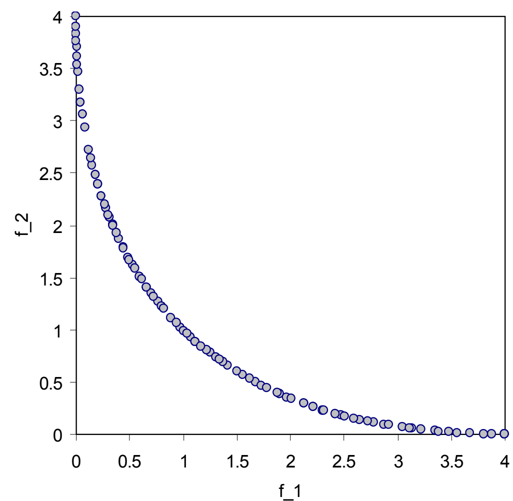
Results for the Two-Objective Test Problem SCH

Although simple, the two-objective test problem SCH (see Table 32 in Appendix A) has a historical significant: almost all proposed MOEAs have been tested using this test problem (Veldhuizen, 1999; Deb, 2001). The test problem SCH has a convex Pareto-optimal front.

The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 80$, average $p_c = 0.7469$, average $\eta_c = 915.73$, average $p_m = 0.9697$, and average $\eta_m = 1940.73$. Figure 12 and diversity metric values for SCH in Table 25 show that the ANSGA-II finds a better spread of Pareto-optimal solutions (distribution of solutions is less crowded in Figure 12(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a smaller population size in solving the problem SCH than the NSGA-II with fixed parameter settings.



(a)



(b)

Figure 12: Non-dominated solutions on SCH with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem FON

The two-objective test problem FON (see Table 32 in Appendix A) has an interesting aspect of arbitrary adding decision variables (scalability) without changing the shape of the Pareto-optimal front and its location in objective space (Veldhuizen, 1999; Deb, 2001). In this study and in the original study of NSGA-II (Deb, Pratap et al., 2002), three decision variables are used (n = 3) in this problem. The test problem FON has a non-convex Pareto-optimal front.

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 80, average p_c = 0.7678, average η_c = 764.17, average p_m = 0.9056, and average η_m = 576.92. Figure 13 and diversity metric values for FON in Table 25 show that the ANSGA-II finds a little bit better spread of Pareto-optimal solutions (distribution of solutions is less crowded in Figure 13(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a smaller population size in solving the problem FON than the NSGA-II with fixed parameter settings.

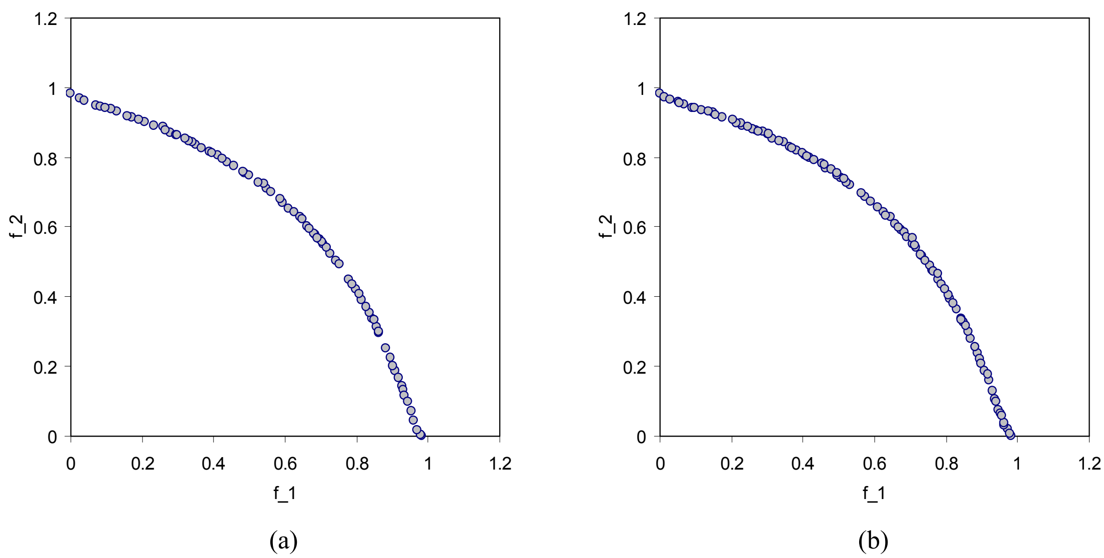
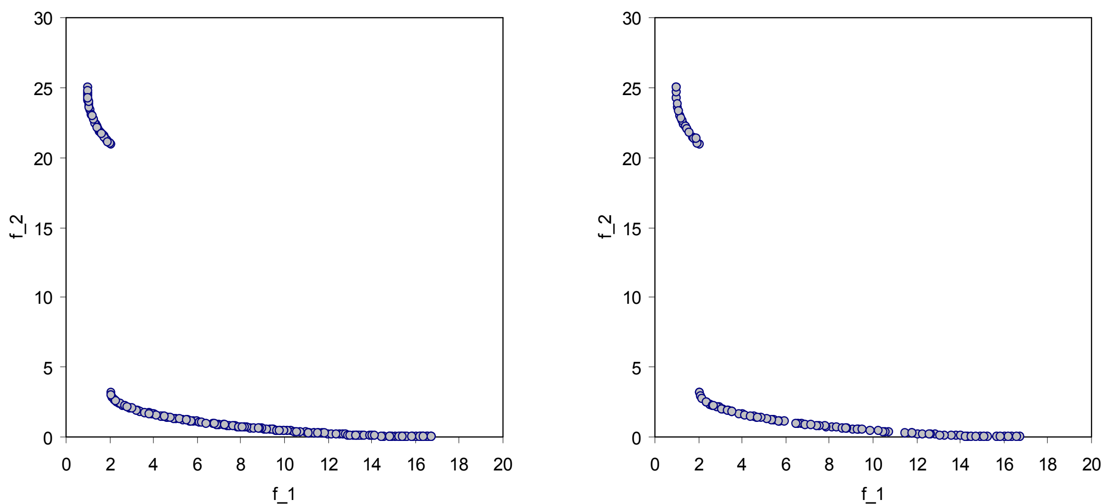


Figure 13: Non-dominated solutions on FON with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem POL

The two-objective test problem POL (see Table 32 in Appendix A) has a non-convex and two disconnected Pareto-optimal fronts. Its solution mapping into objective space appears more complicated than other MOPs from the literature (Veldhuizen, 1999; Deb, 2001). Like other problems having disconnected Pareto-optimal sets, this test problem may cause difficulty to many MOEAs (Deb, 2001).

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 160, average p_c = 0.4740, average η_c = 574.96, average p_m = 0.9255, and average η_m = 143.53. Figure 14 and diversity metric values for POL in Table 25 show that the ANSGA finds a better distribution of Pareto-optimal solutions (distribution of solutions is more continuous on the bottom Pareto-optimal front of Figure 14(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) for the problem POL than the NSGA-II with fixed parameter settings. The ANSGA-II increases the population size to find better solutions ($N = 160$ instead of 100).



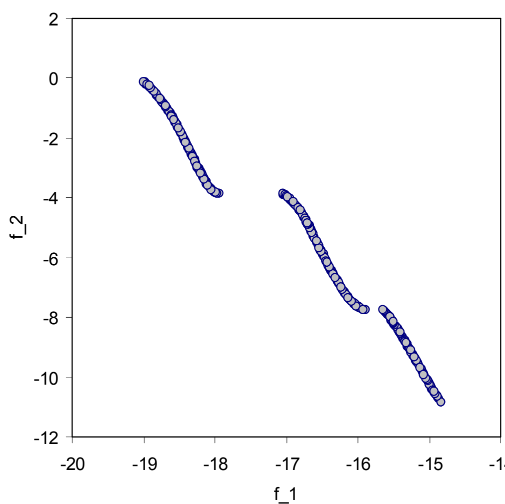
(a) (b)

Figure 14: Non-dominated solutions on POL with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

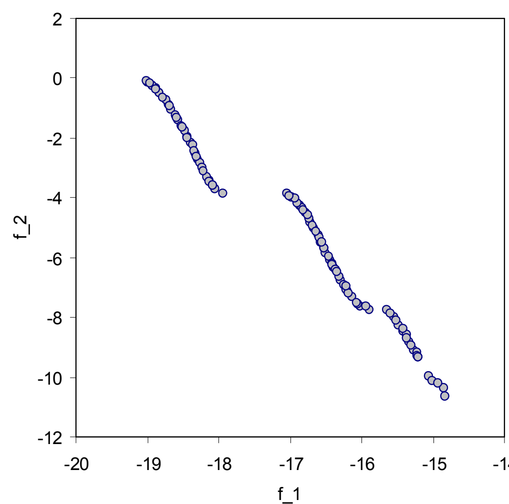
Results for the Two-Objective Test Problem KUR

The two-objective test problem KUR (see Table 32 in Appendix A) has a non-convex and three disconnected Pareto-optimal fronts. Like the problem POL, its solution mapping into objective space is quite complicated. The number of decision variables in this problem can be arbitrary. However, changing the number of decision variables appears to slightly change the shape of its Pareto-optimal front and its location in objective space (Veldhuizen, 1999; Deb, 2001). Like other problems having disconnected Pareto-optimal sets, this test problem may cause difficulty to many MOEAs (Deb, 2001).

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 320, average p_c = 0.8256, average η_c = 482.97, average p_m = 0.5068, and average η_m = 1196.23. Similar to the problem POL, Figure 15 and diversity metric values for KUR in Table 25 show that the ANSGA-II finds a better distribution of Pareto-optimal solutions (distribution of solutions is more continuous on the bottom Pareto-optimal front of Figure 15(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) for the problem KUR using a larger population size than the NSGA-II with fixed parameter settings.



(a)



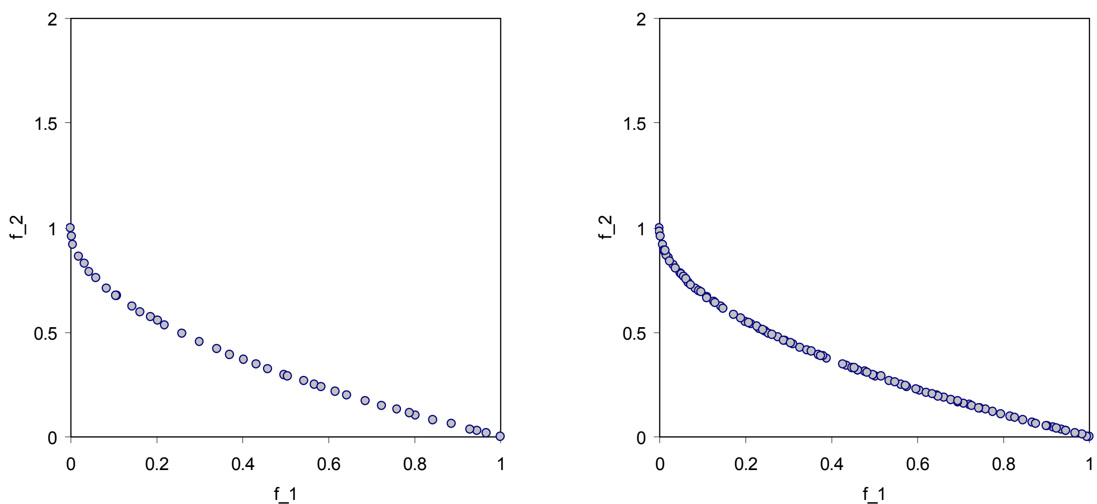
(b)

Figure 15: Non-dominated solutions on KUR with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT1

The two-objective test problem ZDT1 (see Table 32 in Appendix A) has 30 decision variables and a continuous convex Pareto-optimal front. This problem is quite easy. The only difficulty a MOEA may encounter in solving this problem is dealing with a large number of decision variables (Deb, 2001).

The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 40$, average $p_c = 0.6510$, average $\eta_c = 16.43$, average $p_m = 0.8709$, and average $\eta_m = 19.31$. Figure 16 and diversity metric values for ZDT1 in Table 25 show that the ANSGA-II finds a better distribution of Pareto-optimal solutions (distribution of solutions is less crowded and more uniform in Figure 16(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a much smaller population size when compared to the NSGA-II with fixed parameter settings.



(a) (b)

Figure 16: Non-dominated solutions on ZDT1 with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT2

The two-objective test problem ZDT2 (see Table 32 in Appendix A) has 30 decision variables and a continuous non-convex Pareto-optimal front. A MOEA may encounter difficulties in solving this problem because it has to deal with a large number of decision variables and non-convex Pareto-optimal front (Deb, 2001).

The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 40$, average $p_c = 0.0928$, average $\eta_c = 12.29$, average $p_m = 0.6247$, and average $\eta_m = 115.66$. Figure 17 and diversity metric values for ZDT2 in Table 25 show that the ANSGA-II finds a better distribution of Pareto-optimal solutions (distribution of solutions is less crowded and more uniform in Figure 17(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a much smaller population size when compared to the NSGA-II with fixed parameter settings.

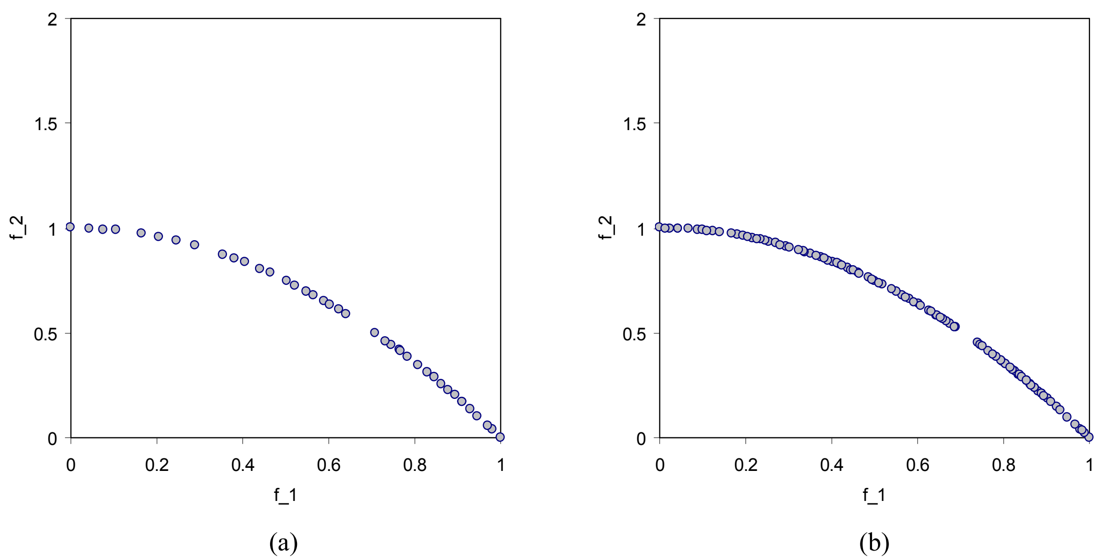
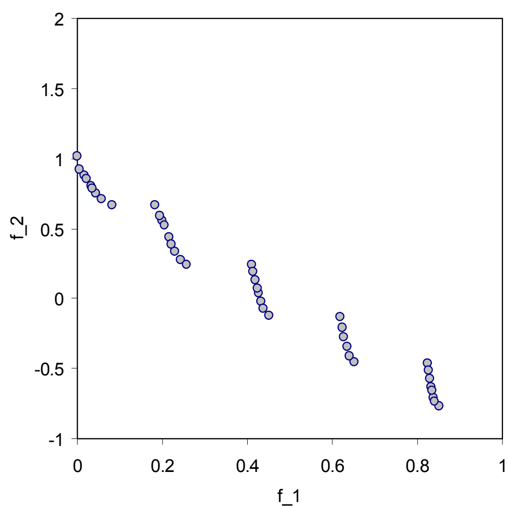


Figure 17: Non-dominated solutions on ZDT2 with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

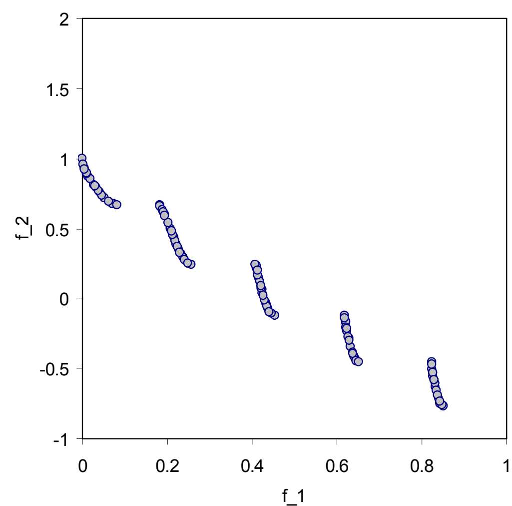
Results for the Two-Objective Test Problem ZDT3

The two-objective test problem ZDT3 (see Table 32 in Appendix A) has 30 decision variables and several disconnected Pareto-optimal fronts. The difficulties a MOEA may encounter in solving this problem include dealing with a large number of decision variables and discontinuous Pareto-optimal fronts. The real test for a MOEA would be to find all discontinuous Pareto-optimal fronts with a uniform spread of non-dominated solutions (Deb, 2001).

The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 40$, average $p_c = 0.0703$, average $\eta_c = 12.68$, average $p_m = 0.7275$, and average $\eta_m = 97.61$. Figure 18 and diversity metric values for ZDT3 in Table 25 show that the ANSGA-II finds a better distribution of Pareto-optimal solutions (distribution of solutions is less crowded and more uniform in Figure 18(a) and $dm_{\text{ANSGA-II}} \le dm_{\text{NSGA-II}} - 0.001$) in solving the problem ZDT3 than the NSGA-II with fixed parameter settings. It is surprised that the ANSGA-II can solve this difficult problem with a very small population size.



(a)



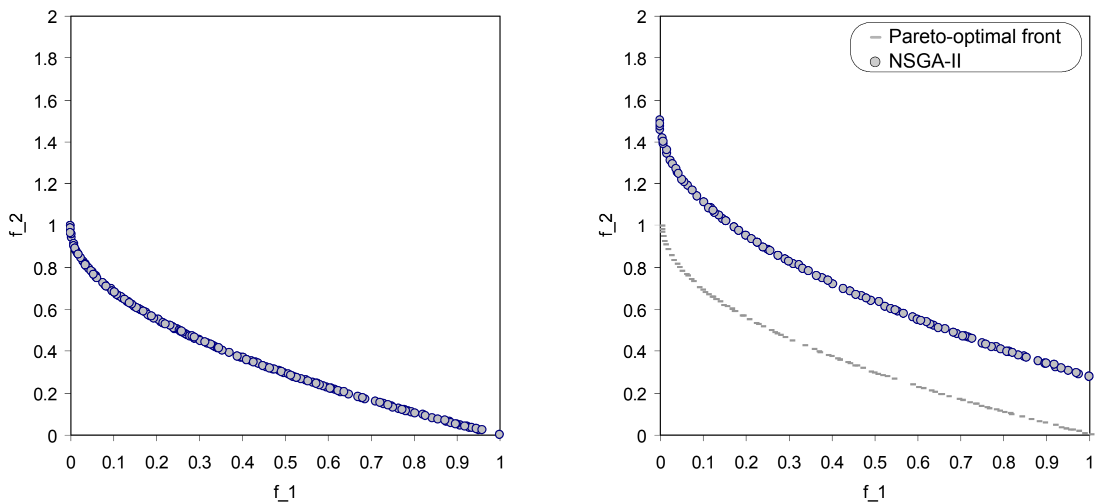
(b)

Figure 18: Non-dominated solutions on ZDT3 with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT4

The two-objective test problem ZDT4 (see Table 32 in Appendix A) has 10 decision variables and a convex Pareto-optimal front. This problem has 21^9 distinct

Pareto-optimal fronts in the objective space, of which only one is global (Deb, 2001). Hence, it may be difficult for a MOEA to converge to the global Pareto-optimal front in solving this problem (Deb, 2001).



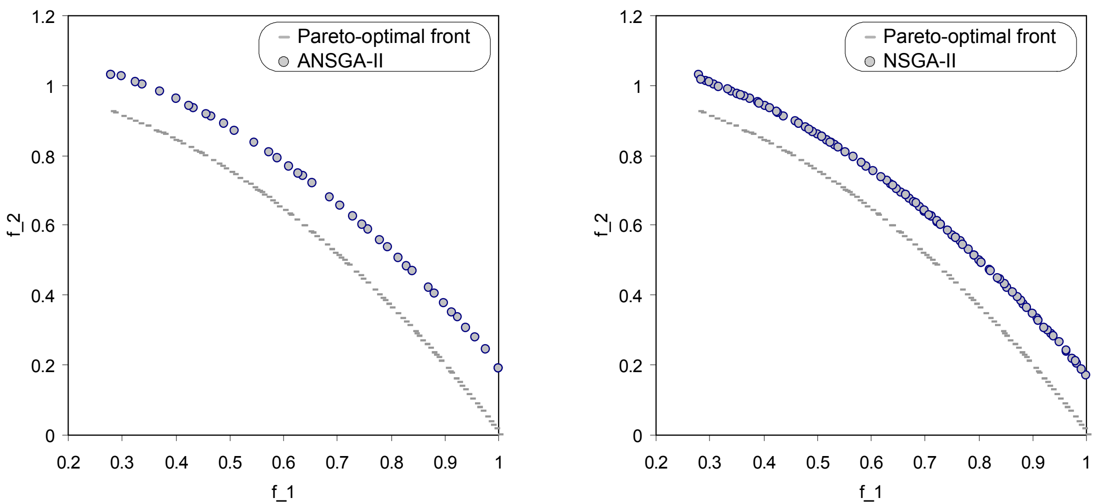
(a) (b)

Figure 19: Non-dominated solutions on ZDT4 with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 160, average p_c = 0.1070, average η_c = 213.88, average p_m = 0.8270, and average η_m = 88.76. Figure 19 shows that the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m is able to find the global Pareto-optimal solutions for the problem ZDT4 while the NSGA-II with fixed parameter settings fails to converge to the global Pareto-optimal front. The ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m requires a larger population size in order to find better solutions for the problem ZDT4. The diversity value of ANSGA-II is larger than that of the NSGA-II for this problem due to more non-dominated solutions found and higher density of solutions in the Pareto-optimal front.

Results for the Two-Objective Test Problem ZDT6

The two-objective test problem ZDT6 (see Table 32 in Appendix A) has 10 decision variables and a non-convex Pareto-optimal front. The density of solutions across the Pareto-optimal front is non-uniform and the density towards the Pareto-optimal front is thin. The non-uniform density of solutions across the Pareto-optimal front combined with the non-convex nature of the front, cause difficulties for many MOEAs to converge to the true Pareto-optimal front in solving this problem (Deb, 2001).



(a) (b)

Figure 20: Non-dominated solutions on ZDT6 with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 40$, average $p_c = 0.0636$, average $\eta_c = 99.94$, average $p_m = 0.3431$, and average $\eta_m = 53.27$. Figure 20 shows that both the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT6. In the original study of NSGA-II, Deb, Pratap et al. (2002) recommended using different parameter settings with the NSGA-II to solve the problem ZDT6. The author of this dissertation carried out the experiment. The NSGA-II was executed with several different parameter settings (such as $N = 200$ to 500, $G = 300$ to 1000, $p_c = 0.5$ to 0.95, $\eta_c = 50$ to 100, $p_m = 0.6$ to 0.9, $\eta_m = 10$ to 2000) in solving ZDT6. However, the NSGA-II still fails to converge to the true Pareto-optimal front for this problem. It is obvious that the issue is rooted in the NSGA-II and it may have to do with the way the NSGA-II handling elitism (as described in Section "Barriers and Issues" in Chapter 1). The author of this dissertation recommends a further investigation on this issue. Figure 20 and diversity metric values in Table 25 show that the ANSGA-II produces a better spread of non-dominated solutions (distribution of solutions is less crowded and more uniform in Figure 20(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) than the NSGA-II. The ANSGA-II requires much smaller population size in solving this problem.

Results for the Two-Objective Test Problem DEB

The two-objective constraint test problem DEB (see Table 33 in Appendix A) has a convex Pareto-optimal front. Constraints divide the search space into two regions – feasible and infeasible regions. However, the entire Pareto-optimal front of this problem is still convex. Constraints may cause difficulty for MOEAs to converge to the true Pareto-optimal front and to maintain a diverse set of non-dominated solutions (Deb, 2001).

The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 80$, average $p_c = 0.9187$, average $\eta_c = 696.30$, average $p_m = 0.8756$, and average $\eta_m = 367.75$. Figure 21 and diversity metric values for DEB in Table 25 show that the ANSGA-II finds a better distribution of Pareto-optimal solutions (distribution of solutions is less crowded in Figure 21(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a smaller population size when compared to the NSGA-II with fixed parameter settings.

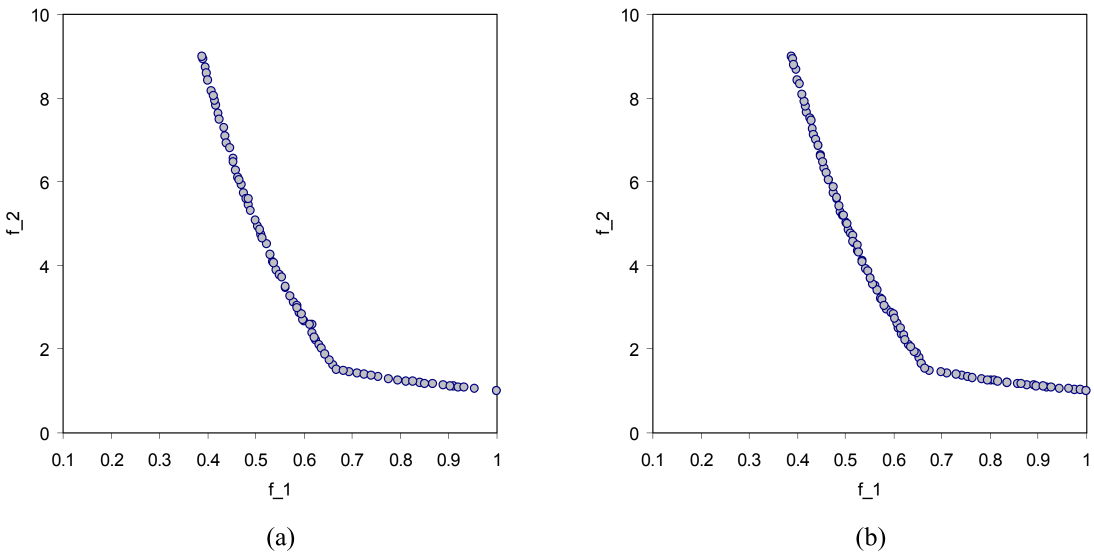
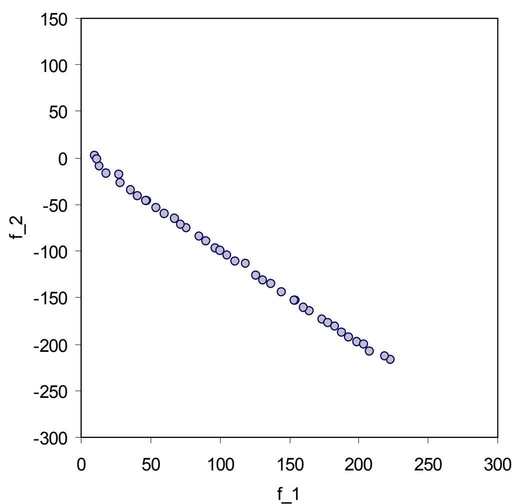


Figure 21: Non-dominated solutions on DEB with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

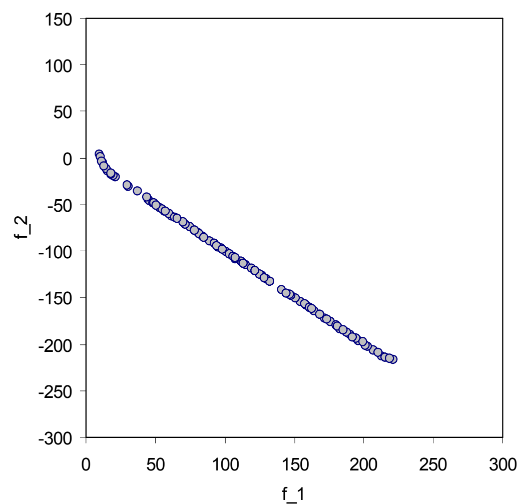
Results for the Two-Objective Test Problem SRN

The two-objective constraint test problem SRN (see Table 33 in Appendix A) has a convex Pareto-optimal front. Constraints divide the search space into two regions – feasible and infeasible regions. The constraints of this problem eliminate some regions of the unconstrained Pareto-optimal set, which may cause difficulty for MOEAs to converge to the true Pareto-optimal front and to maintain a diverse set of non-dominated solutions (Deb, 2001).

The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 40$, average $p_c = 0.0971$, average $\eta_c = 661.05$, average $p_m = 0.8829$, and average $\eta_m = 136.66$. Figure 22(a) shows that convergence is adequate with the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m because one non-dominated solution does not line up on the true Pareto-optimal front. However, diversity metric values for SRN in Table 25 shows that the ANSGA-II achieves a better distribution of solutions ($dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) on the problem SRN using a much smaller population size than the NSGA-II. Therefore, it can be said that the performance of two algorithms is equivalent. The ANSGA-II requires less time to find the Pareto-optimal solutions for this problem than the NSGA-II. This implies that the overhead for learning good parameter values for the problem SRN is minimum.



(a)

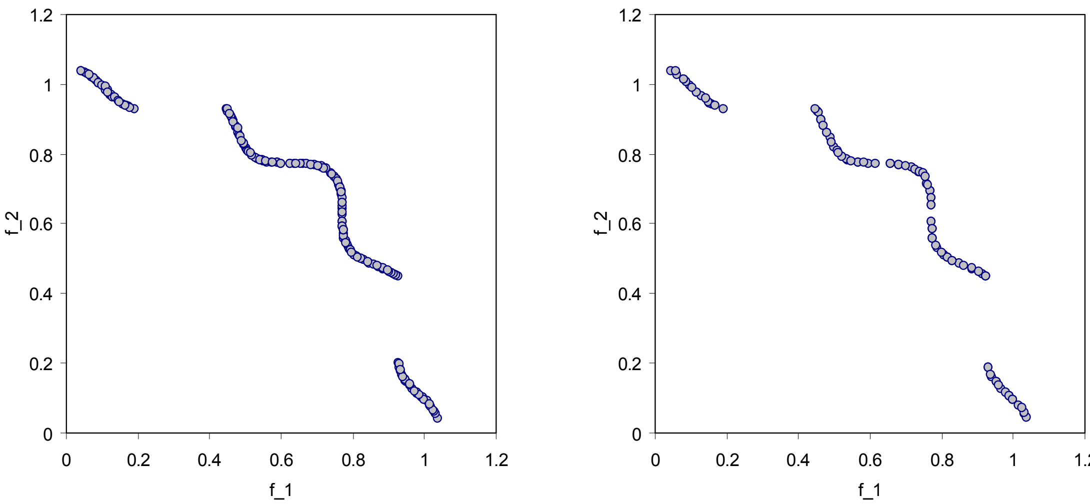


(b)

Figure 22: Non-dominated solutions on SRN with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem TNK

The two-objective constraint test problem TNK (see Table 33 in Appendix A) has a non-convex and three disconnected Pareto-optimal fronts. Constraints divide the search space into two regions – feasible and infeasible regions. The Pareto-optimal solutions of this problem lie on a non-linear surface (see Figure 23), which may cause difficulty for MOEAs to converge to the true Pareto-optimal fronts and to maintain a diverse set of non-dominated solutions across all three discontinuous Pareto-optimal fronts (Deb, 2001).



(a) (b)

Figure 23: Non-dominated solutions on TNK with (a) ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 160, average p_c = 0.4614, average η_c = 361.16, average p_m = 0.8059, and average η_m = 1371.04. Figure 23 shows that the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m finds a more continuous Pareto-optimal front in the middle than the NSGA-II with fixed parameter settings. Table 25 also shows that the ANSGA-II requires a larger population size to find better solutions for the problem TNK. The diversity value of ANSGA-II is little bit larger than that of the NSGA-II for this problem due to more non-dominated solutions found and higher density of solutions on the Pareto-optimal front.

Results for the Five-Objective Real-World Problem WATER

The constraint real-world problem WATER (see Table 33 in Appendix A) has five objectives and seven constraints. For a large number of objectives, the Pareto-optimal set has multi-dimensional in the objective space. Hence, MOEAs need better strategies for finding a diverse set of non-dominated solutions and approximating to the true Pareto-optimal front with a reasonable computational effort.

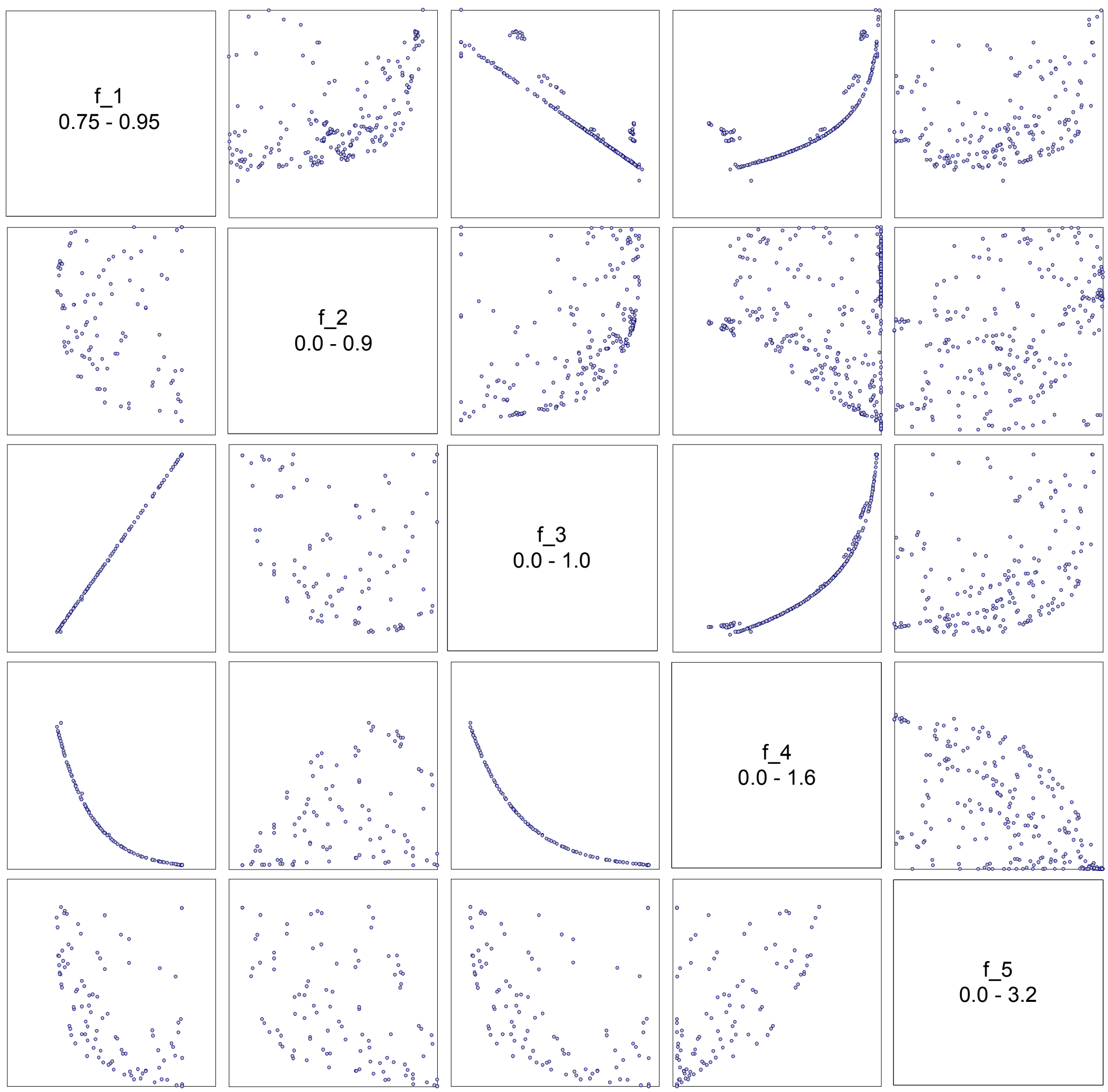


Figure 24: Non-dominated solutions on WATER with upper diagonal plots for ANSGA-II (adaptable N, p_c, p_m, η_c, η_m) and lower diagonal plots for NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 400, average p_c = 0.3029, average η_c = 128.56, average p_m = 0.9694, and average η_m = 78.72. Figure 24 and Table 25 show that the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m finds more non-dominated solutions for the problem WATER than the NSGA-II with fixed parameter settings. The plots for ANSGA-II have better formed patterns than the plots for NSGA-II, implying that the ANSGA-II achieves better convergence than NSGA-II (i.e. plots for f_1-f_2, f_1-f_5, f_2-f_3, f_2-f_4, f_3-f_5, f_4-f_5). Table 25 also shows that the ANSGA-II requires a larger population size in order to find better solutions for the problem WATER. The diversity value of ANSGA-II is a little bit larger than that of the NSGA-II for this problem due to more solutions found, higher density of solutions in the Pareto-optimal front, and some generated solutions are not in the Pareto-optimal front.

Results of ANSGA-II with Adaptable N, p_c, η_c, and Fixed p_m, η_m

This variant of ANSGA-II supports adaptive population size (N), self-adaptive crossover probability (p_c), and self-adaptive crossover distribution index (η_c). The mutation probability (p_m) and mutation distribution index (η_m) are set to the same parameter values used in the NSGA-II: p_m = 0.5 and η_m = 100. Table 26 presents performance results of the ANSGA-II with adaptable N, p_c, η_c against the original NSGA-II with fixed parameter settings on thirteen benchmark problems.

Table 26: Performance results of ANSGA-II adaptable (N, p_c, η_c, and fixed p_m, η_m) against the original NSGA-II with fixed parameter settings

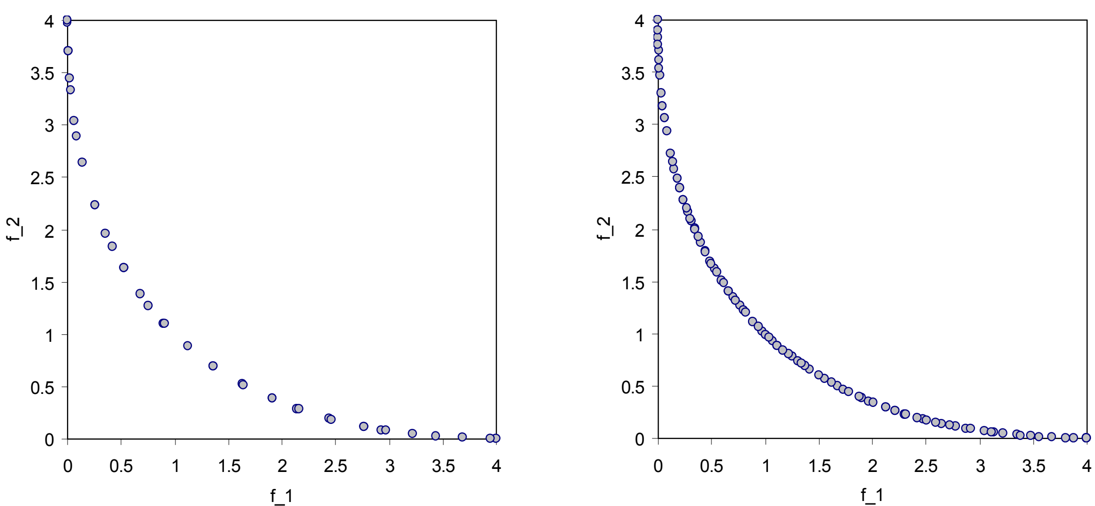
Problems	ANSGA-II						NSGA-II					
	N	G	Diversity Metric	Func. Eval.	Time (sec)	N*G (1000)	N	G	Diversity Metric	Func. Eval.	Time (sec)	N*G (1000)
SCH	40	288	0.7199	709	12	11.52	100	250	0.5711	251	4	25
FON	40	256	0.7270	515	11	10.24	100	250	0.7219	251	4	25

POL	80	340	1.1829	767	26	27.2	100	250	0.9538	251	3	25
KUR	160	352	0.7901	927	63	56.32	100	250	0.8121	251	4	25
ZDT1	40	320	0.7235	809	21	12.8	100	250	0.7431	251	4	25
ZDT2	320	384	0.7026	1080	167	122.88	100	250	0.7390	251	6	25
ZDT3	40	240	0.6972	499	36	9.6	100	250	0.8731	251	5	25
ZDT4	80	376	0.5302	966	66	30.08	100	250	0.6766	251	5	25
ZDT6	40	612	0.7592	612	15	24.48	100	250	0.7046	251	8	25
DEB	40	284	0.7989	543	9	11.36	100	500	0.7772	501	15	50
SRN	40	284	0.7765	798	27	11.36	100	500	0.8011	501	16	50
TNK	80	444	0.7806	1015	39	35.52	100	500	0.8072	501	8	50
WATER	200	526	0.7301	1169	393	105.2	100	500	0.6274	501	10	50

For the problems KUR, ZDT2, and TNK, the ANSGA-II with adaptable N, p_c, η_c out-performs the original NSGA-II with fixed parameter settings in terms of finding better distribution of non-dominated solutions and approximating to the true Pareto-optimal front. For problems with many local Pareto-optimal fronts (e.g. ZDT4), the ANSGA-II with adaptable N, p_c, η_c fails to achieve good convergence. This variant of ANSGA-II also performs poorly on problems POL, DEB, and WATER. Both ANSGA-II and NSGA-II fail to converge to the true Pareto-optimal front on the problem ZDT6. For other problems in the suite of thirteen benchmark multi-objective test problems used in this study (SCH, FON, ZDT1, ZDT3, SRN), the ANSGA-II with adaptable N, p_c, η_c achieves satisfactory convergence and the distribution of non-dominated solutions is nearly the same to that of the NSGA-II with fixed parameter settings. This variant of the ANSGA-II requires less time to solve the problem DEB than the NSGA-II. For other problems, this variant is slower than the NSGA-II due to overheads, which appear to be acceptable, of solving the problem and learning good parameter values at the same time. The plots of non-dominated solutions on thirteen benchmark problems obtained by the ANSGA-II and the NSGA-II are presented and compared in the following.

Results for the Two-Objective Test Problem SCH

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_c = 0.4445, and average η_c = 292.8. Figure 25 and diversity metric values for SCH in Table 26 show that the ANSGA-II with adaptable N, p_c, η_c finds a less spread of Pareto-optimal solutions (less solutions on the Pareto-optimal front of Figure 25(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) using a much smaller population size in solving the problem SCH than the NSGA-II with fixed parameter settings. However, the distribution of Pareto-optimal solutions with ANSGA-II is adequate ($dm_{\text{ANSGA-II}} \leq 0.80$).



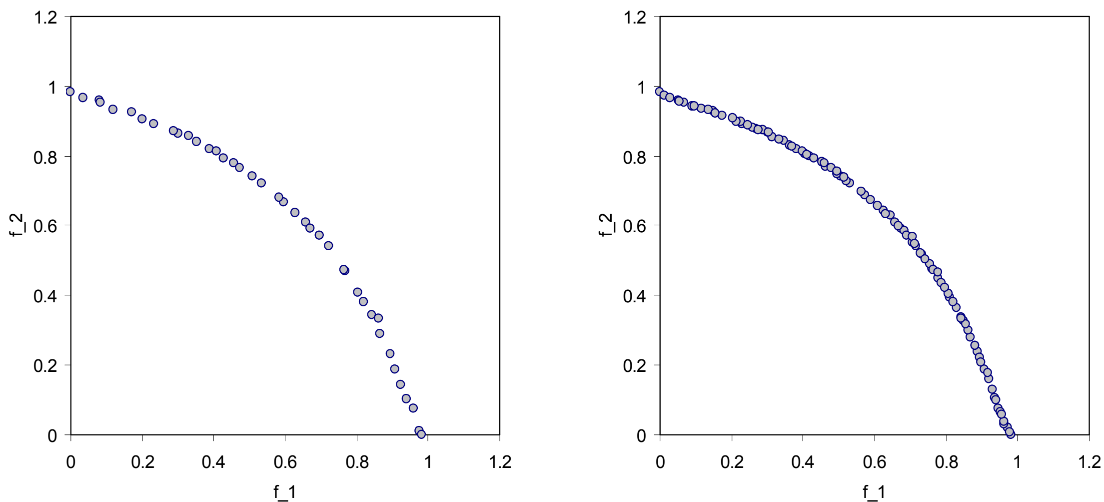
(a) (b)

Figure 25: Non-dominated solutions on SCH with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem FON

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_c = 0.7692, and average η_c = 646.84. Figure 26 and diversity metric values for FON in Table 26 show that this variant finds a less spread of Pareto-optimal solutions (less solutions on the Pareto-optimal front of Figure 26(a) and

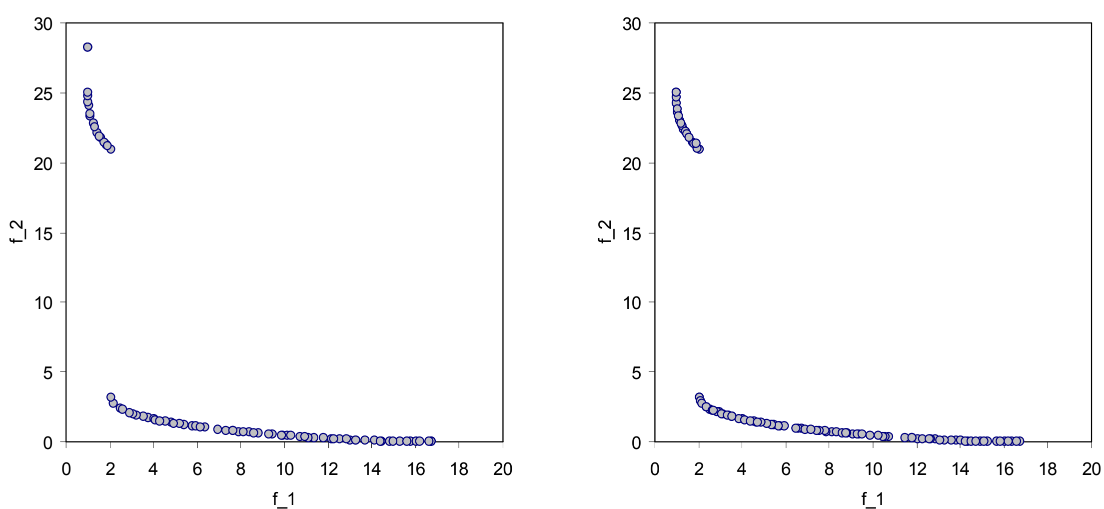
$dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) using much smaller population size in solving the problem FON compared to the NSGA-II with fixed parameter settings. However, the distribution of Pareto-optimal solutions with ANSGA-II is adequate ($dm_{\text{ANSGA-II}} \leq 0.80$).



(a) (b)

Figure 26: Non-dominated solutions on FON with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem POL

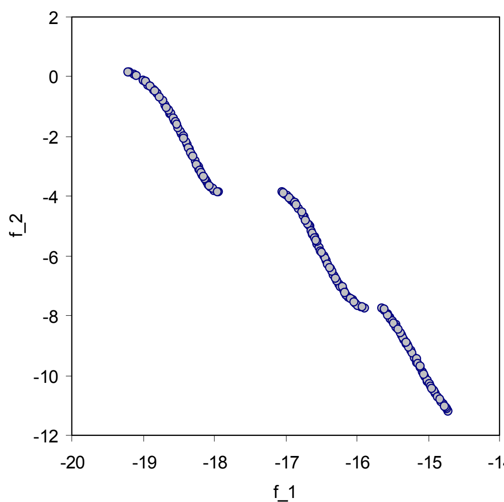


(a) (b)

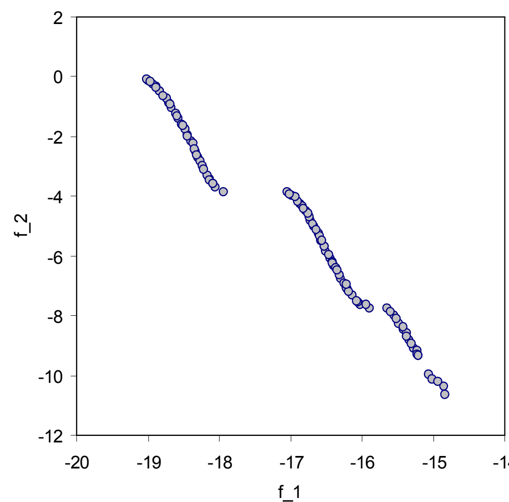
Figure 27: Non-dominated solutions on POL with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 80, average p_c = 0.1017, and average η_c = 683.29. Figure 27(a) shows the ANSGA-II with adaptable N, p_c, η_c performs worse than the NSGA-II because it generates one non-dominated solution that is not in the true Pareto-optimal front. Its diversity metric value in Table 26, which is greater than one, also reflects this issue ($dm_{\text{ANSGA-II}}$ = 1.1829).

Results for the Two-Objective Test Problem KUR



(a)



(b)

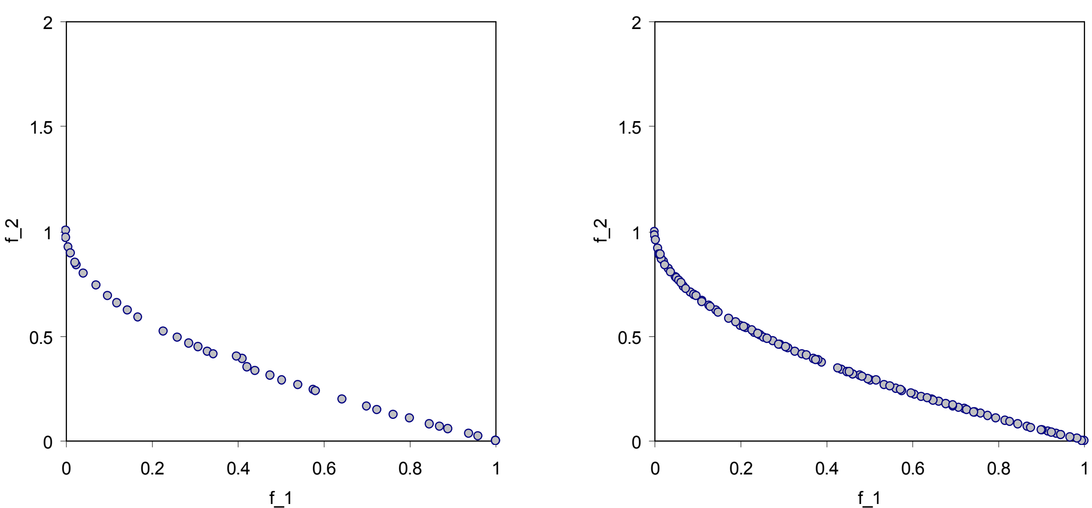
Figure 28: Non-dominated solutions on KUR with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 160, average p_c = 0.5880, and average η_c = 389.32. Figure 28 and diversity metric values for KUR in Table 26 show that the ANSGA-II with adaptable N, p_c, η_c finds a better spread of Pareto-optimal solutions (distribution of solutions is more continuous on the bottom Pareto-optimal front of Figure 28(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}}$ – 0.001) in solving the problem KUR than the NSGA-II with fixed parameter settings.

The algorithm requires a much larger population size in order to find better solutions for this problem. However, the required population size is smaller than that of the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m for this problem ($N = 320$).

Results for the Two-Objective Test Problem ZDT1

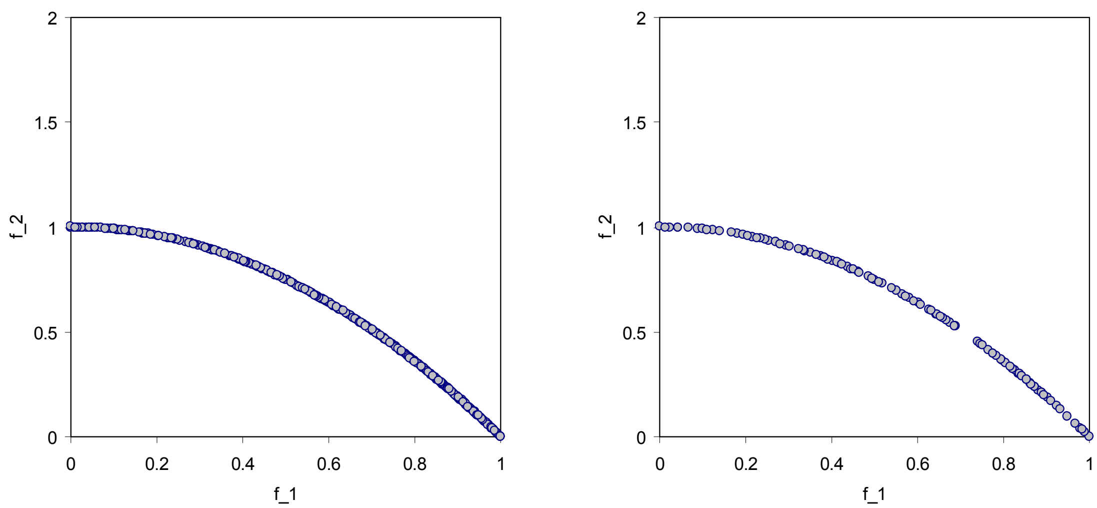
The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_c = 0.2077, and average η_c = 83.32. Figure 29 and diversity metric values for ZDT1 in Table 26 show that the ANSGA-II with adaptable N, p_c, η_c finds a better spread of non-dominated solutions (distribution of solutions is less crowded and more uniform in Figure 29(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) with adequate convergence (two non-dominated solutions do not line up on the true Pareto-optimal front) in solving the problem ZDT1 compared to the NSGA-II with fixed parameter settings.



(a) (b)

Figure 29: Non-dominated solutions on ZDT1 with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT2



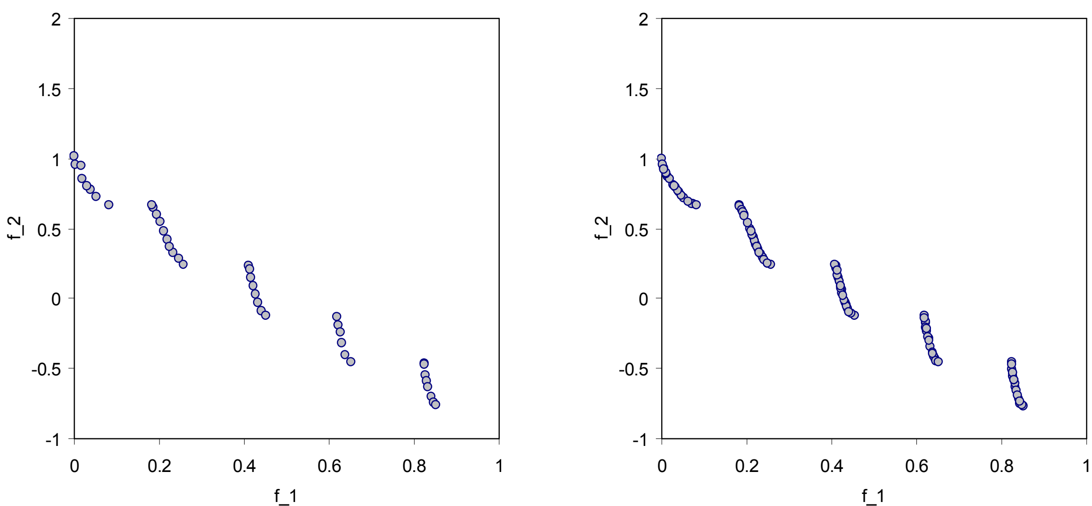
(a) (b)

Figure 30: Non-dominated solutions on ZDT2 with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 320$, average $p_c = 0.0947$, and average $\eta_c = 310.33$. Figure 30 and diversity metric values for ZDT2 in Table 26 show that the ANSGA-II with adaptable N, p_c, η_c finds a better spread of Pareto-optimal solutions (distribution of solutions is more continuous on the Pareto-optimal front of Figure 30(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem ZDT2 compared to the NSGA-II with fixed parameter settings. The ANSGA-II requires a much larger population size in order to find better solutions for this problem than that of both the NSGA-II and the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m. This implies that the fixed settings ($p_m = 0.5$, $\eta_m = 100$) do not provide enough diversity of candidate solutions in the population for this problem (the average $p_m = 0.6247$ and the average $\eta_m = 115.66$ as found by the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m on this problem). Therefore, this variant of ANSGA-II takes more time to find proper values for these parameters.

Results for the Two-Objective Test Problem ZDT3

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_c = 0.1286, and average η_c = 132.96. Figure 31 and diversity metric values for ZDT3 in Table 26 show that the ANSGA-II with adaptable N, p_c, η_c finds a better spread of non-dominated solutions (distribution of solutions is less crowded and more uniform in Figure 31(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) with adequate convergence (one non-dominated solution does not line up on the true Pareto-optimal front) in solving the problem ZDT3 compared to the NSGA-II with fixed parameter settings. The algorithm requires a much smaller population size to solve this problem.



(a) (b)

Figure 31: Non-dominated solutions on ZDT3 with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT4

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 80, average p_c = 0.6045, and average η_c = 788.43. Figure 32 shows that both the ANSGA-II with adaptable N, p_c, η_c and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT4. However, the NSGA-II approximates closer to the global Pareto-optimal front. The most likely reason for the ANSGA-II to fail converging to the true Pareto-optimal front is that there is not enough diversity in the population but the distribution of solutions is good ($dm_{\text{ANSGA-II}} \leq 0.80$ as shown in Table 26); therefore, the algorithm terminates prematurely with a population size $N = 80$.

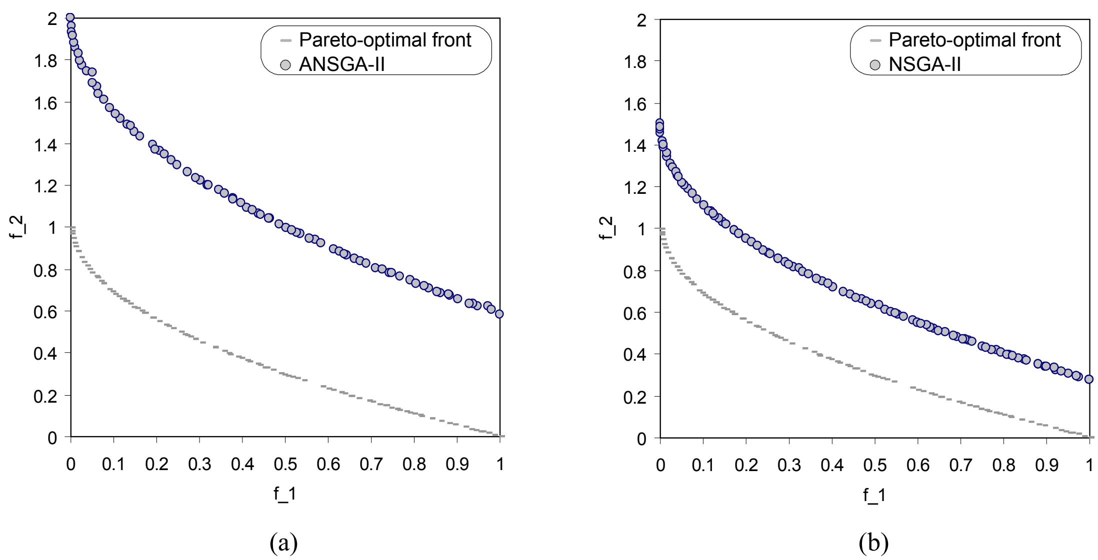
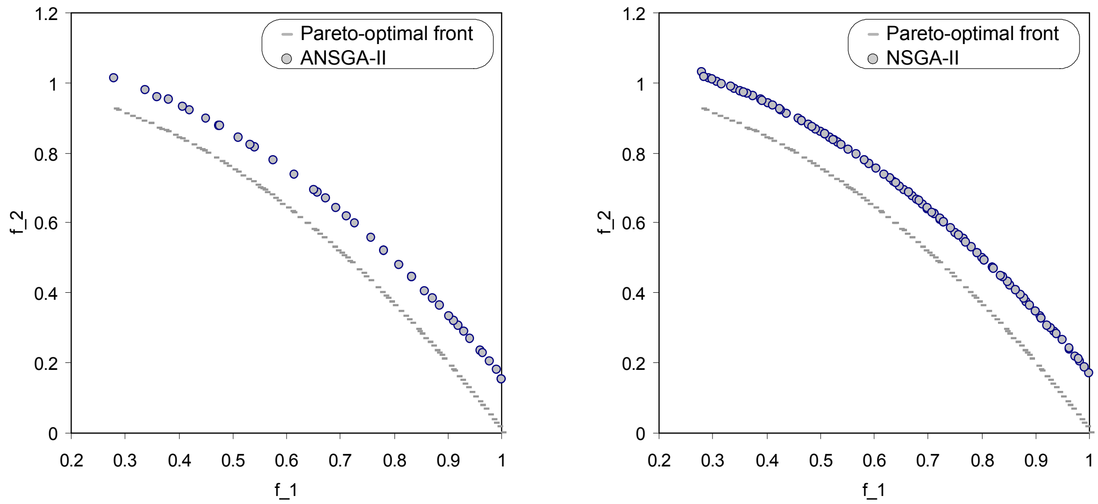


Figure 32: Non-dominated solutions on ZDT4 with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT6

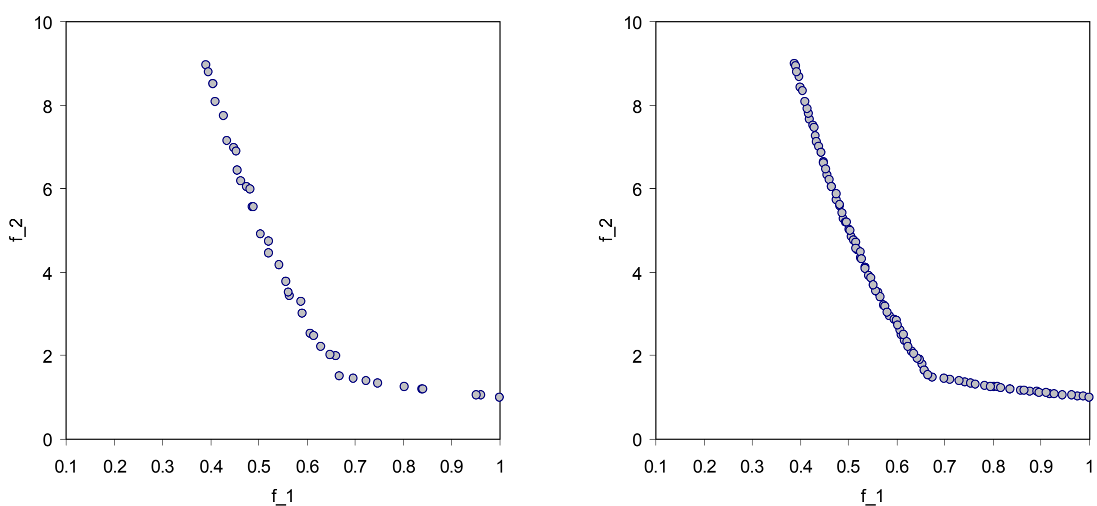
The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 40$, average $p_c = 0.94$, and average $\eta_c = 67.67$. Similar to the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m, Figure 33 shows that both the ANSGA-II with adaptable parameters N, p_c, η_c and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT6.



(a) (b)

Figure 33: Non-dominated solutions on ZDT6 with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem DEB

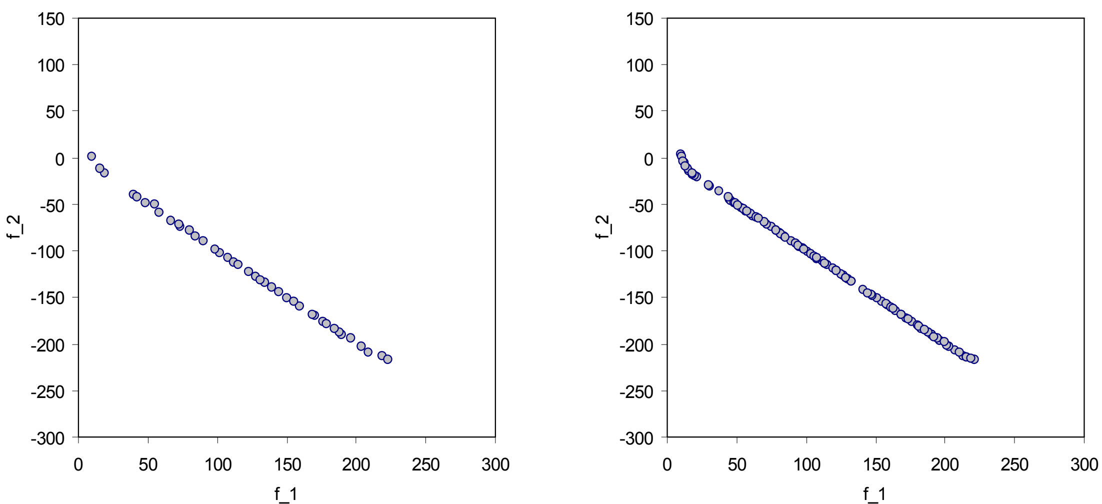


(a) (b)

Figure 34: Non-dominated solutions on DEB with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_c = 0.623, and average η_c = 583.47. Figure 34 and diversity metric values for DEB in Table 26 show that the ANSGA-II with adaptable parameters N, p_c, η_c finds a less converged and less spread set of non-dominated solutions (distribution of solutions is less uniform in Figure 34(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) for problem DEB than the NSGA-II with fixed parameter settings. However, the diversity value of ANSGA-II is good enough ($dm_{\text{ANSGA-II}} \leq 0.80$); therefore, the algorithm terminates prematurely with a population size $N = 40$. It can be said this variant performs adequately on this problem.

Results for the Two-Objective Test Problem SRN



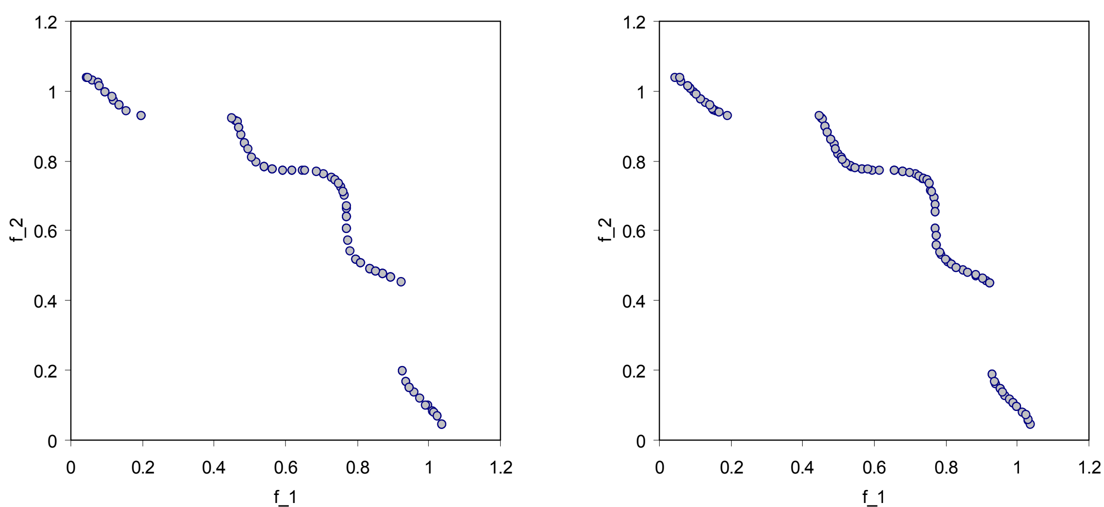
(a) (b)

Figure 35: Non-dominated solutions on SRN with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: $N = 40$, average $p_c = 0.5497$, and average $\eta_c = 631.85$. Figure 35 shows that the ANSGA-II with adaptable N, p_c, η_c finds a less uniform spread of Pareto-optimal solutions in solving the problem SRN compared to the NSGA-II with fixed parameter settings. However, Table 26 shows that the diversity metric value of the ANSGA-II is better than that of the NSGA-II because the NSGA-II generates more condensed solutions in the Pareto-optimal front on this problem. This variant of ANSGA-II requires a much smaller population size and less time to solve this problem than the NSGA-II.

Results for the Two-Objective Test Problem TNK

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 80, average p_c = 0.6065, and average η_c = 378.47. Figure 36 and diversity metric values for TNK in Table 26 show that the ANSGA-II with adaptable N, p_c, η_c finds a better spread of Pareto-optimal solutions (distribution of solutions is less crowded and more continuous in the middle Pareto-optimal front of Figure 36(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a smaller population size to solve this problem than the NSGA-II with fixed parameter settings.



(a) (b)

Figure 36: Non-dominated solutions on TNK with (a) ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Five-Objective Real-World Problem WATER

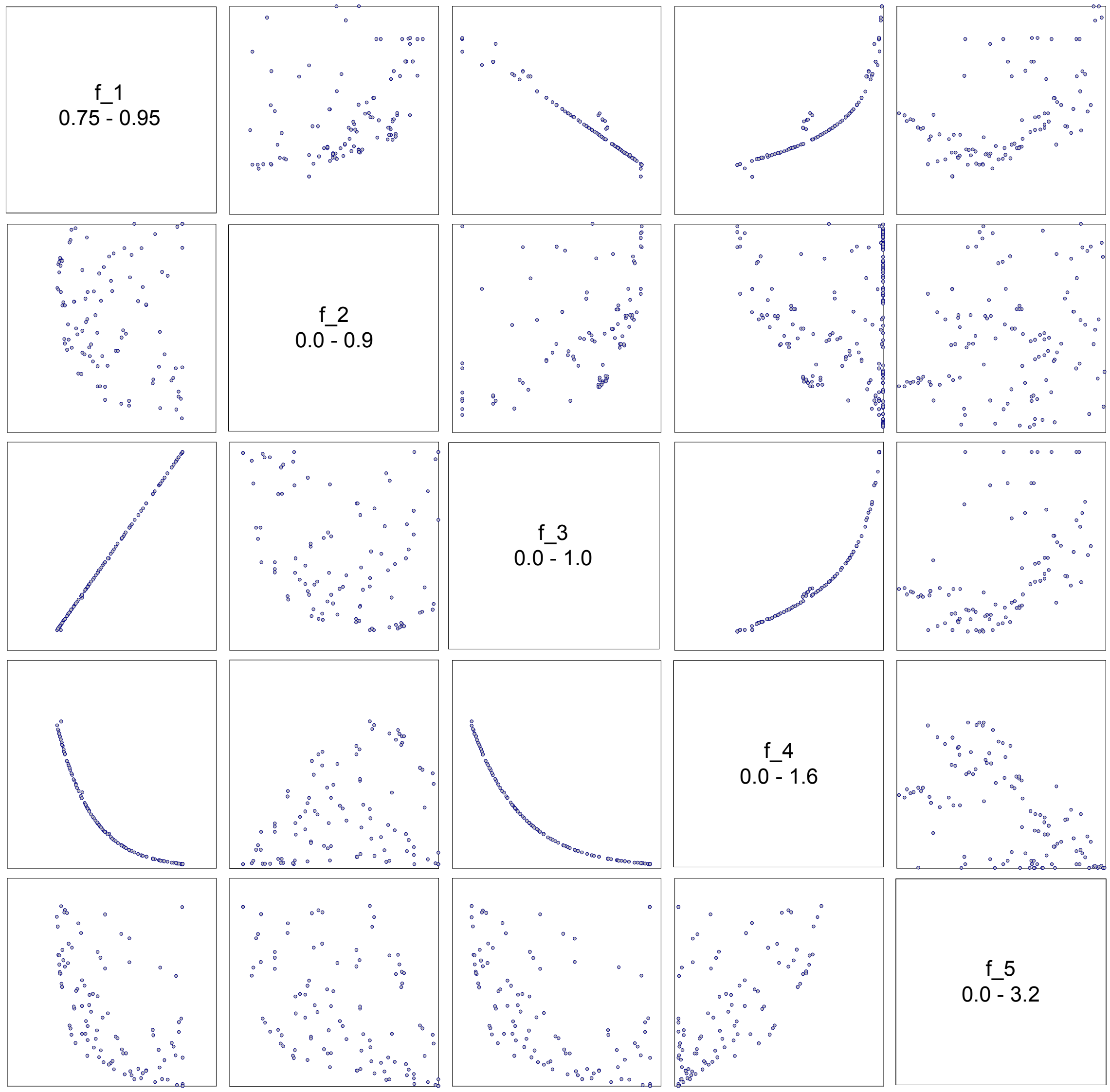


Figure 37: Non-dominated solutions on WATER with upper diagonal plots for ANSGA-II (adaptable N, p_c, η_c, and fixed p_m, η_m) and lower diagonal plots for NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 200, average p_c = 0.3694, and average η_c = 108.56. Figure 37 and diversity values for WATER in Table 26 show that the ANSGA-II with adaptable parameters N, p_c, η_c finds a less uniform spread of non-dominated solutions (less distribution of solutions on the Pareto-optimal fronts in upper diagonal plots of Figure 37 and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) for the problem WATER than the NSGA-II with fixed parameter settings. The plots for ANSGA-II have less formed-patterns than the plots for NSGA-II, implying that this variant of ANSGA-II achieves less convergence than NSGA-II on this complex problem. This variant of ANSGA-II also requires a larger population size in order to solve this problem.

Results of ANSGA-II with Adaptable N, p_m, η_m, and Fixed p_c, η_c

This variant of ANSGA-II supports adaptive population size (N), self-adaptive mutation probability (p_m), and self-adaptive mutation distribution index (η_m). The crossover probability (p_c) and crossover distribution index (η_c) are set to the same parameter values used in the NSGA-II: p_c = 0.9 and η_c = 20. Table 27 presents performance results of the ANSGA-II with adaptable N, p_m, η_m against the original NSGA-II with fixed parameter settings on thirteen benchmark problems.

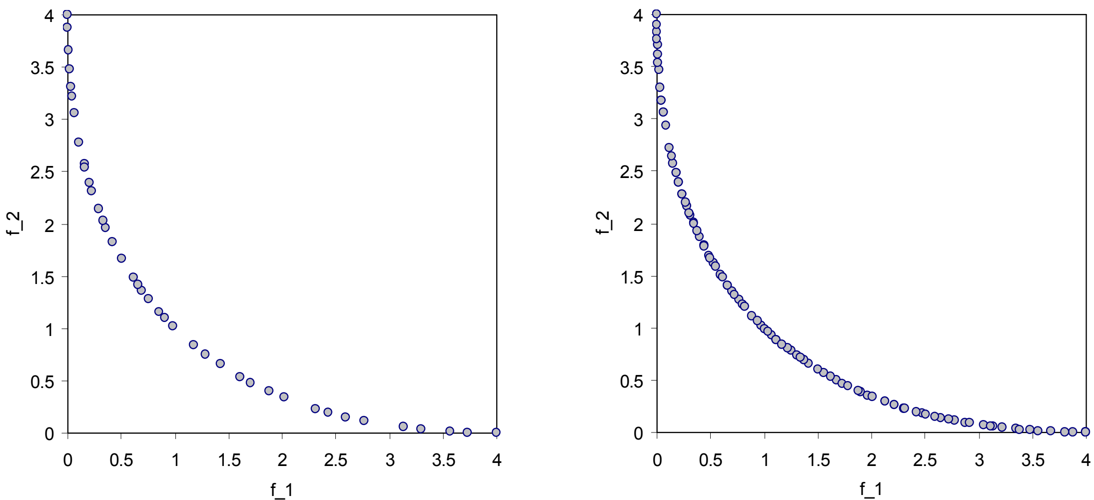
Table 27: Performance results of ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) against the original NSGA-II with fixed parameter settings

Problems	ANSGA-II						NSGA-II					
	N	*G*	Diversity Metric	Func. Eval.	Time (sec)	*N*G* (1000)	*N*	*G*	Diversity Metric	Func. Eval.	Time (sec)	*N*G* (1000)
SCH	40	246	0.5285	525	8	9.84	100	250	0.5711	251	4	25
FON	40	280	0.7079	595	12	11.2	100	250	0.7219	251	4	25
POL	80	352	0.5904	710	34	28.16	100	250	0.9538	251	3	25
KUR	160	368	0.8392	867	57	58.88	100	250	0.8121	251	4	25
ZDT1	40	308	0.7227	741	15	12.32	100	250	0.7431	251	4	25
ZDT2	40	322	0.6573	634	17	12.88	100	250	0.7390	251	6	25
ZDT3	40	304	0.7650	733	15	12.16	100	250	0.8731	251	5	25
ZDT4	40	320	0.5715	732	16	12.8	100	250	0.6766	251	5	25
ZDT6	80	372	0.7283	855	45	29.76	100	250	0.7046	251	8	25
DEB	80	416	0.7867	899	29	33.28	100	500	0.7772	501	15	50
SRN	40	256	0.7870	487	9	10.24	100	500	0.8011	501	16	50
TNK	800	1728	0.7415	6224	1978	1382.4	100	500	0.8072	501	8	50
WATER	800	560	0.6639	1043	519	448	100	500	0.6274	501	10	50

For the problems SCH, POL, ZDT1, ZDT3, and SRN, the ANSGA-II with adaptable N, p_m, η_m out-performs the original NSGA-II with fixed parameter settings in terms of finding better distribution of non-dominated solutions and approximating to the true Pareto-optimal front. For the problem WATER, it achieves better convergence but a little bit less distribution of solutions than the NSGA-II. This variant of ANSGA-II achieves satisfactory convergence and the distribution of non-dominated solutions is nearly the same to that of the NSGA-II on two problems FON and DEB. This variant of ANSGA-II also achieves good convergence and better distribution of non-dominated solutions on the problem TNK than the NSGA-II; however, it takes unsatisfactory long execution time on this problem (1978 seconds). For other problems (KUR, ZDT2, ZDT4, ZDT6), the NSGA-II with fixed parameter settings performs better. This variant of the ANSGA-II requires less time to solve the problem SRN than the NSGA-II. For other problems, this variant is slower than the NSGA-II due to overheads, which appear to be acceptable (except for problem TNK), of solving the problem and learning good parameter values at the same time. The plots of non-dominated solutions on thirteen benchmark problems obtained by the ANSGA-II and the NSGA-II are presented and compared in the following.

Results for the Two-Objective Test Problem SCH

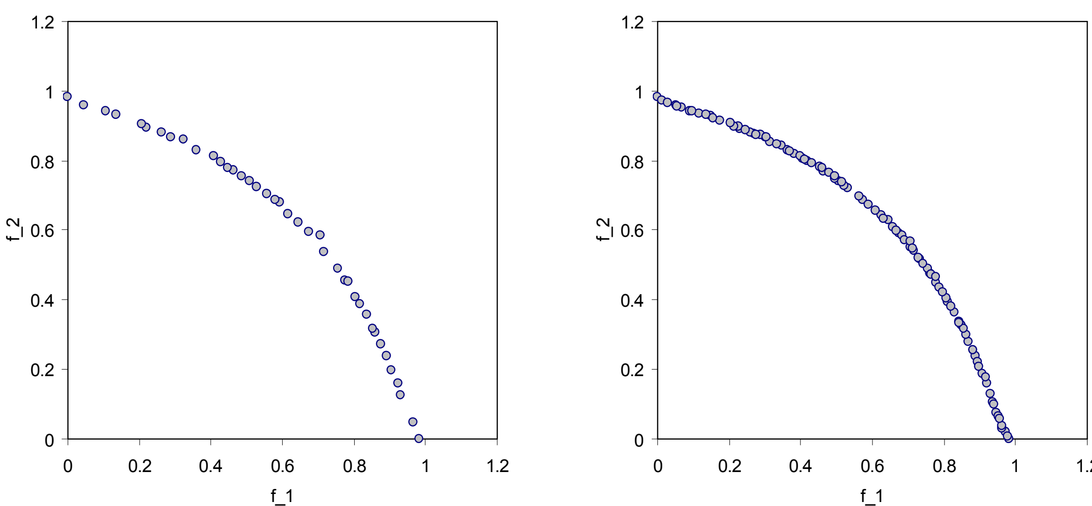
The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_m = 0.9445, and average η_m = 474.94. Figure 38 and diversity metric values for SCH in Table 27 show that the ANSGA-II with adaptable N, p_m, η_m finds a better spread of Pareto-optimal solutions (distribution of solutions is less crowded and more uniform in Figure 38(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a much smaller population size in solving the problem SCH than the NSGA-II with fixed parameter settings.



(a) (b)

Figure 38: Non-dominated solutions on SCH with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem FON

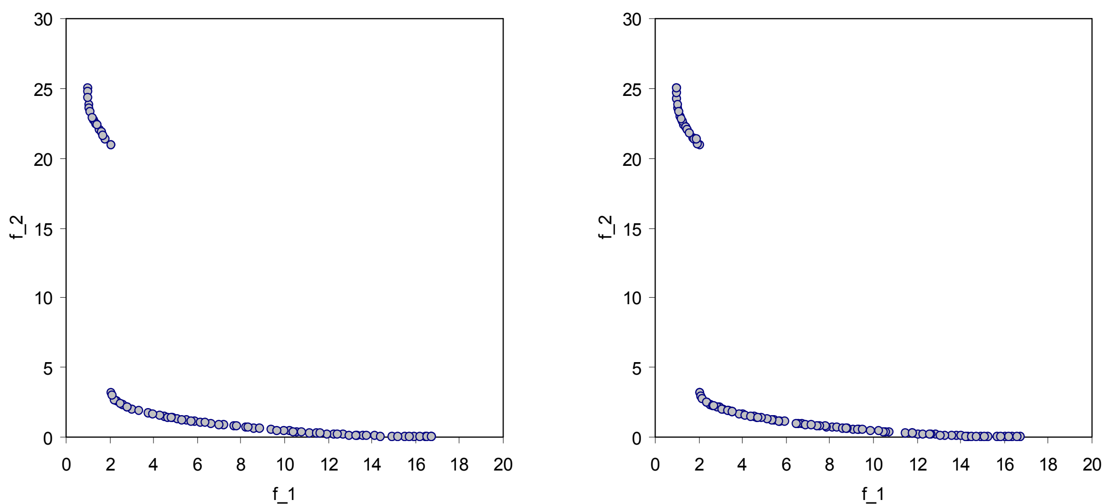


(a) (b)

Figure 39: Non-dominated solutions on FON with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_m = 0.8390, and average η_m = 220.12. Figure 39 and diversity metric values for FON in Table 27 show that the ANSGA-II with adaptable N, p_m, η_m finds a better spread of non-dominated solutions (distribution of solutions is less crowded and more uniform in Figure 39(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) with adequate convergence (one non-dominated solution does not line up on the true Pareto-optimal front) compared to the NSGA-II with fixed parameter settings. The ANSGA-II uses a much smaller population size to solve this easy problem than the NSGA-II.

Results for the Two-Objective Test Problem POL



(a) (b)

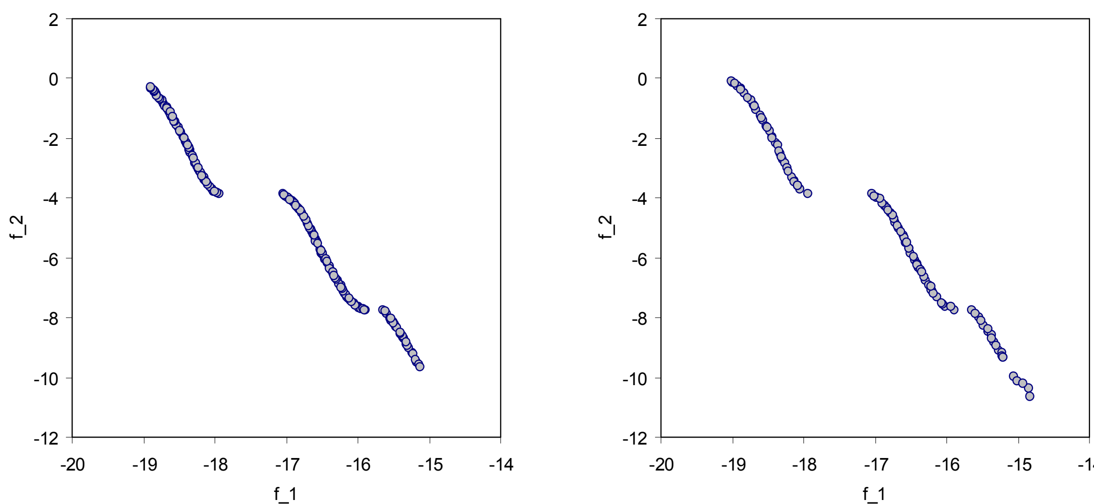
Figure 40: Non-dominated solutions on POL with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 80, average p_m = 0.7831, and average η_m = 328.23. Figure 40 and diversity metric values for POL in Table 27 show that this variant finds a better spread of Pareto-optimal solutions (distribution of solutions is less crowded and more continuous in

Figure 40(a), and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$). The ANSGA-II requires a smaller population size but a larger number of generations to solve this problem than the NSGA-II. As a result, the number of solutions evaluated to find the non-dominated set is higher than that of required by the NSGA-II.

Results for the Two-Objective Test Problem KUR

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 160, average p_m = 0.5684, and average η_m = 1280.11. Figure 41(a) shows that the ANSGA-II with adaptable N, p_m, η_m performs worse than the NSGA-II with fixed parameter settings because it fails to obtain non-dominated solutions that cover the entire shape of the Pareto-optimal front (bottom right region).

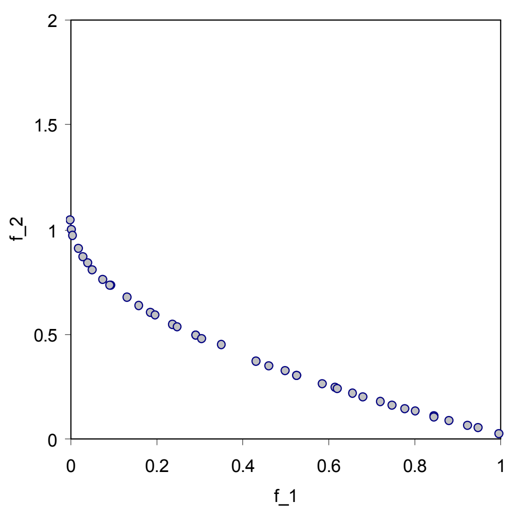


(a) (b)

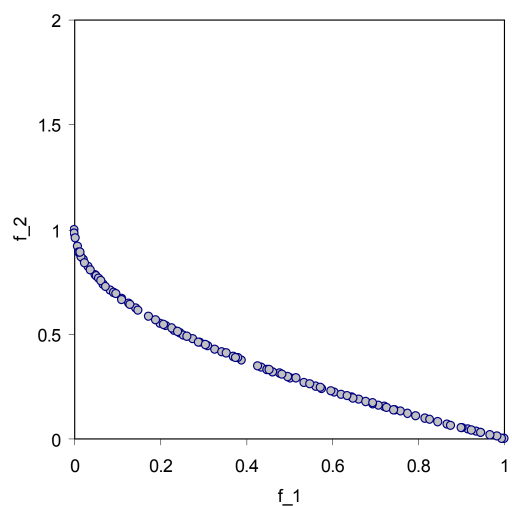
Figure 41: Non-dominated solutions on KUR with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT1

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_m = 0.0192, and average η_m = 66.96. Figure 42 and diversity metric values for ZDT1 in Table 27 show that the ANSGA-II with adaptable N, p_m, η_m finds a better spread of Pareto-optimal solutions (distribution of solutions is less crowded and more uniform in Figure 42(a) and $dm_{\text{ANSGA-II}} \le dm_{\text{NSGA-II}} - 0.001$) using a smaller population size in solving the problem ZDT1 than the NSGA-II with fixed parameter settings.



(a)

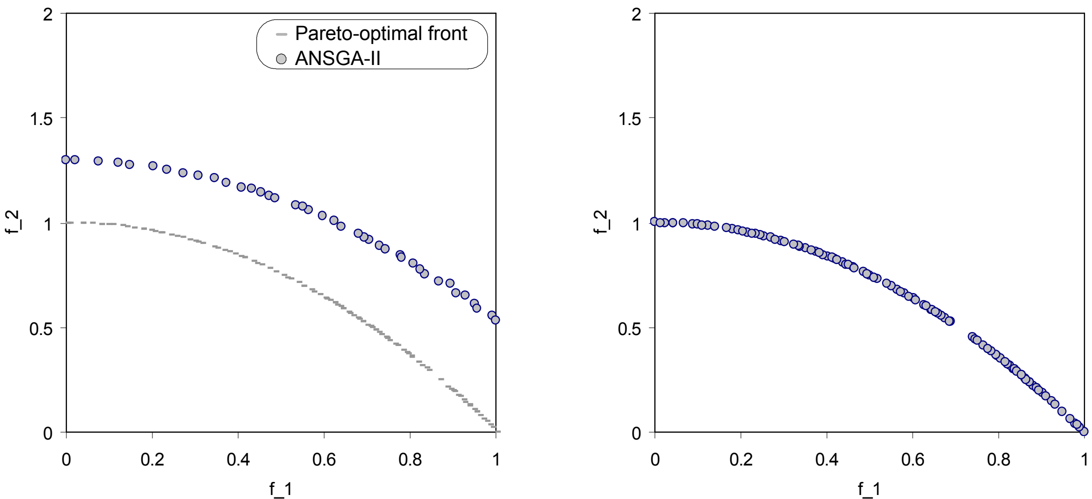


(b)

Figure 42: Non-dominated solutions on ZDT1 with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT2

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_m = 0.0837, and average η_m = 724.96. Figure 43 shows that the ANSGA-II with adaptable N, p_m, η_m performs worse than the NSGA-II because it fails to converge to the global Pareto-optimal front in solving the problem ZDT2. The most likely reason for the ANSGA-II to fail converging to the true Pareto-optimal front is that there is not enough diversity of candidate solutions in the population but the distribution of solutions is good ($dm_{\text{ANSGA-II}} = 0.6573$) as shown in Table 27. Therefore, the algorithm terminates prematurely with a population size $N = 40$.

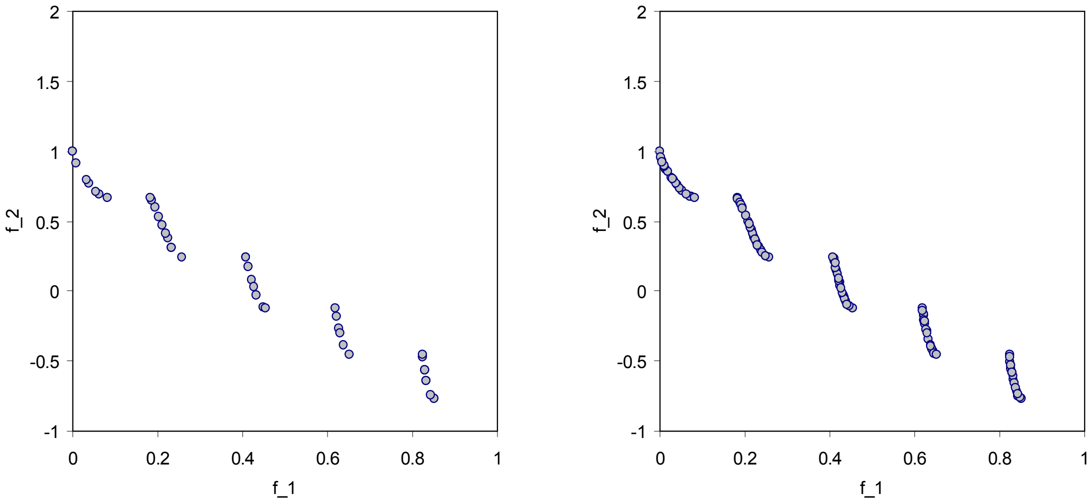


(a) (b)

Figure 43: Non-dominated solutions on ZDT2 with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT3

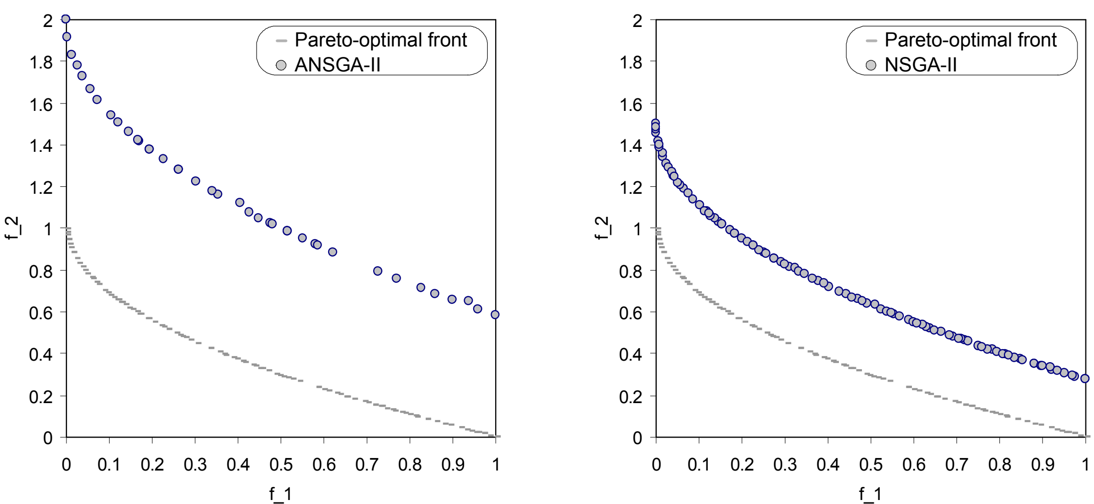
The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_m = 0.0369, and average η_m = 38.59. Figure 44 and diversity metric values for ZDT3 in Table 27 show that the ANSGA-II with adaptable N, p_m, η_m finds a better spread of Pareto-optimal solutions (distribution of solutions is less crowded and more uniform in Figure 44(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a much smaller population size in solving the problem ZDT3 than the NSGA-II with fixed parameter settings.



(a) (b)

Figure 44: Non-dominated solutions on ZDT3 with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT4



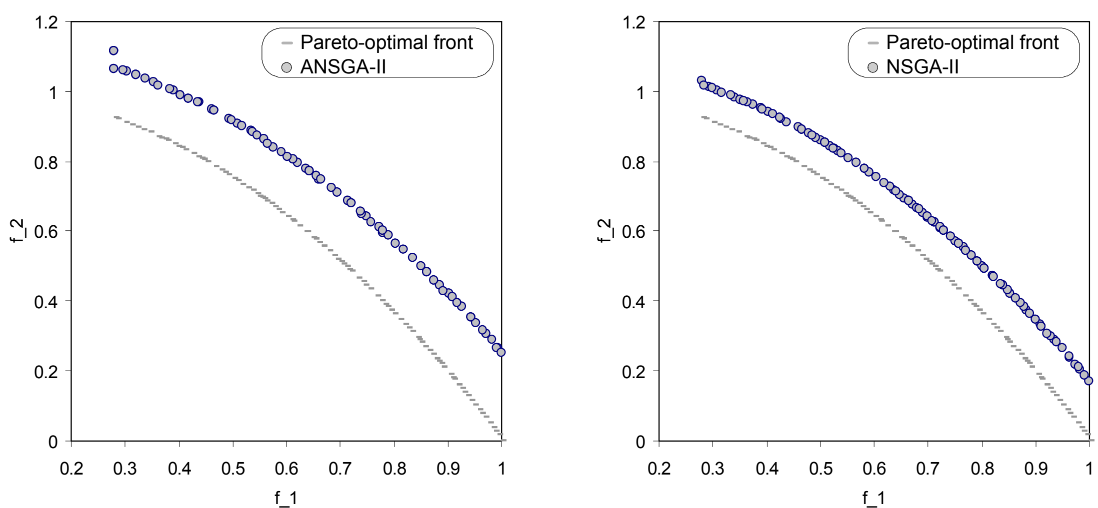
(a) (b)

Figure 45: Non-dominated solutions on ZDT4 with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_m = 0.1170, and average η_m = 220.85. Figure 45 shows that both the ANSGA-II with adaptable N, p_m, η_m and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT4. However, the NSGA-II approximates closer to the global Pareto-optimal front than the ANSGA-II.

Results for the Two-Objective Test Problem ZDT6

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 80, average p_m = 0.0158, and average η_m = 13.82. Similar to the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m, Figure 46 shows that both the ANSGA-II with adaptable parameters N, p_m, η_m and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT6. However, the NSGA-II approximates closer to the global Pareto-optimal front. Figure 46(a) also shows that this variant of ANSGA-II generates one non-dominated solution that does not line up on the local Pareto-optimal front.



(a) (b)

Figure 46: Non-dominated solutions on ZDT6 with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem DEB

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 80, average p_m = 0.8724, and average η_m = 543.64. Figure 47 and diversity metric values for DEB in Table 27 show that the ANSGA-II with adaptable N, p_m, η_m finds a less spread of Pareto-optimal solutions (less uniform distribution of solutions on the Pareto-optimal front in Figure 47(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) using a much smaller population size in solving the problem DEB than the NSGA-II with fixed parameter settings. However, the distribution of Pareto-optimal solutions with ANSGA-II is adequate ($dm_{\text{ANSGA-II}} \leq 0.80$).

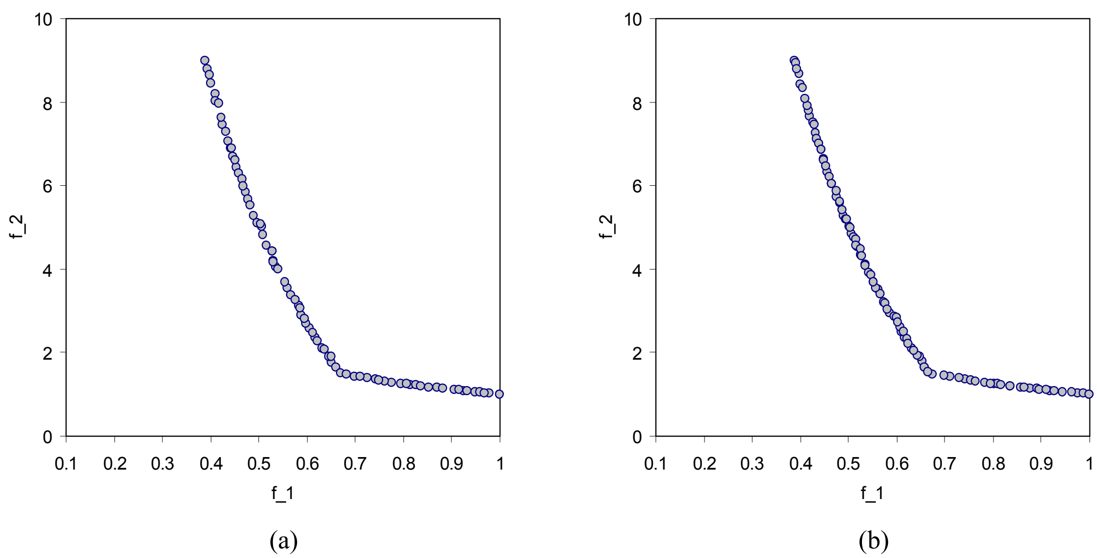
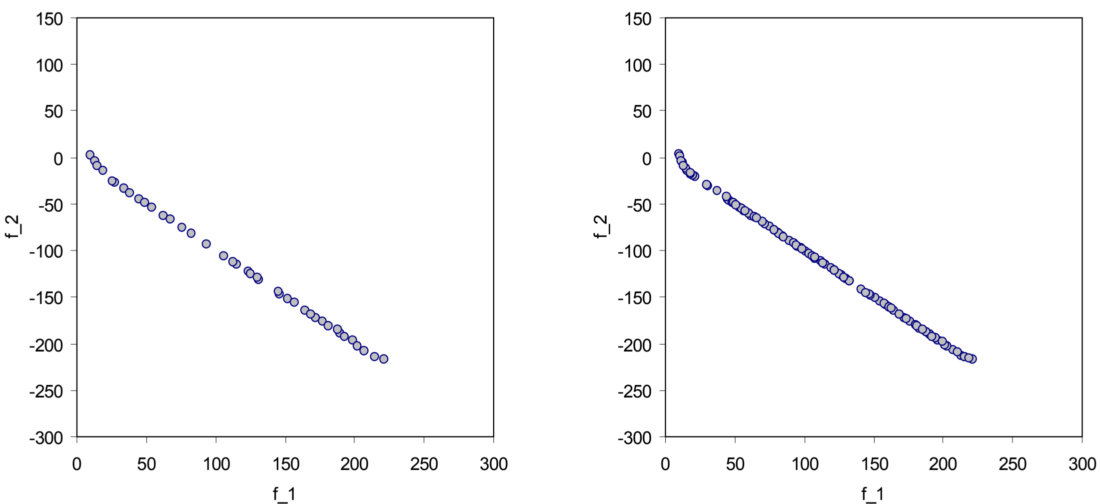


Figure 47: Non-dominated solutions on DEB with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem SRN

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 40, average p_m = 0.8907, and average η_m = 342.89. Figure 48 and diversity metric values for SRN in Table 27 show that the ANSGA-II with adaptable N,

p_m, η_m finds a better spread of Pareto-optimal solutions (distribution of solutions is less crowded and more uniform in Figure 48(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a smaller population size in solving the problem SRN than the NSGA-II with fixed parameter settings.

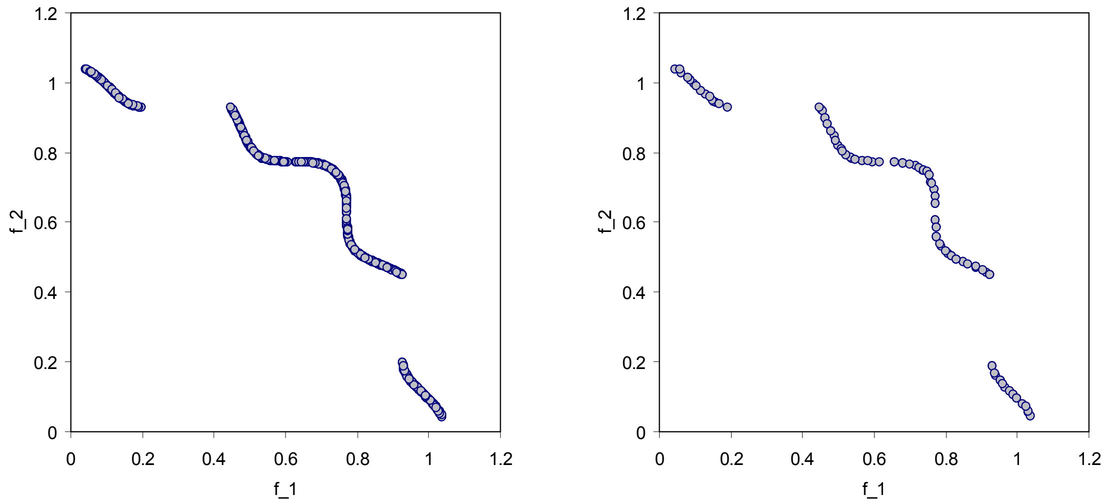


(a) (b)

Figure 48: Non-dominated solutions on SRN with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem TNK

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 800, average p_m = 0.2281, and average η_m = 1713.22. Figure 49 and diversity metric values for TNK in Table 27 show that the ANSGA-II with adaptable N, p_m, η_m finds a better spread of Pareto-optimal solutions (distribution of solutions is more continuous on the middle Pareto-optimal front of Figure 49(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem TNK than the NSGA-II with fixed parameter settings. However, this variant of ANSGA-II requires a much bigger population size and it takes an unacceptable long time to solve this problem (1078 seconds).



(a) (b)

Figure 49: Non-dominated solutions on TNK with (a) ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Five-Objective Real-World Problem WATER

The values of the adaptable parameters identified by the ANSGA-II for this problem are: N = 800, average p_m = 0.8826, and average η_m = 280.81. Figure 50 and diversity metric values for WATER in Table 27 show that the ANSGA-II with adaptable parameters N, p_m, η_m finds an a less spread of non-dominated solutions (less distribution of solutions on the Pareto-optimal fronts in upper diagonal plots of Figure 50 and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) for the problem WATER compared to that of the NSGA-II with fixed parameter settings. However, the distribution of Pareto-optimal solutions with ANSGA-II is adequate ($dm_{\text{ANSGA-II}} \leq 0.80$). The plots for ANSGA-II have better formed patterns than the plots for NSGA-II, implying that the ANSGA-II achieves better convergence than NSGA-II (e.g. plots for f_1-f_2, f_1-f_5, f_2-f_3, f_2-f_4, f_3-f_5, f_4-f_5). Table 27 also shows that the ANSGA-II requires larger population size in order to find more non-dominated solutions for this problem.

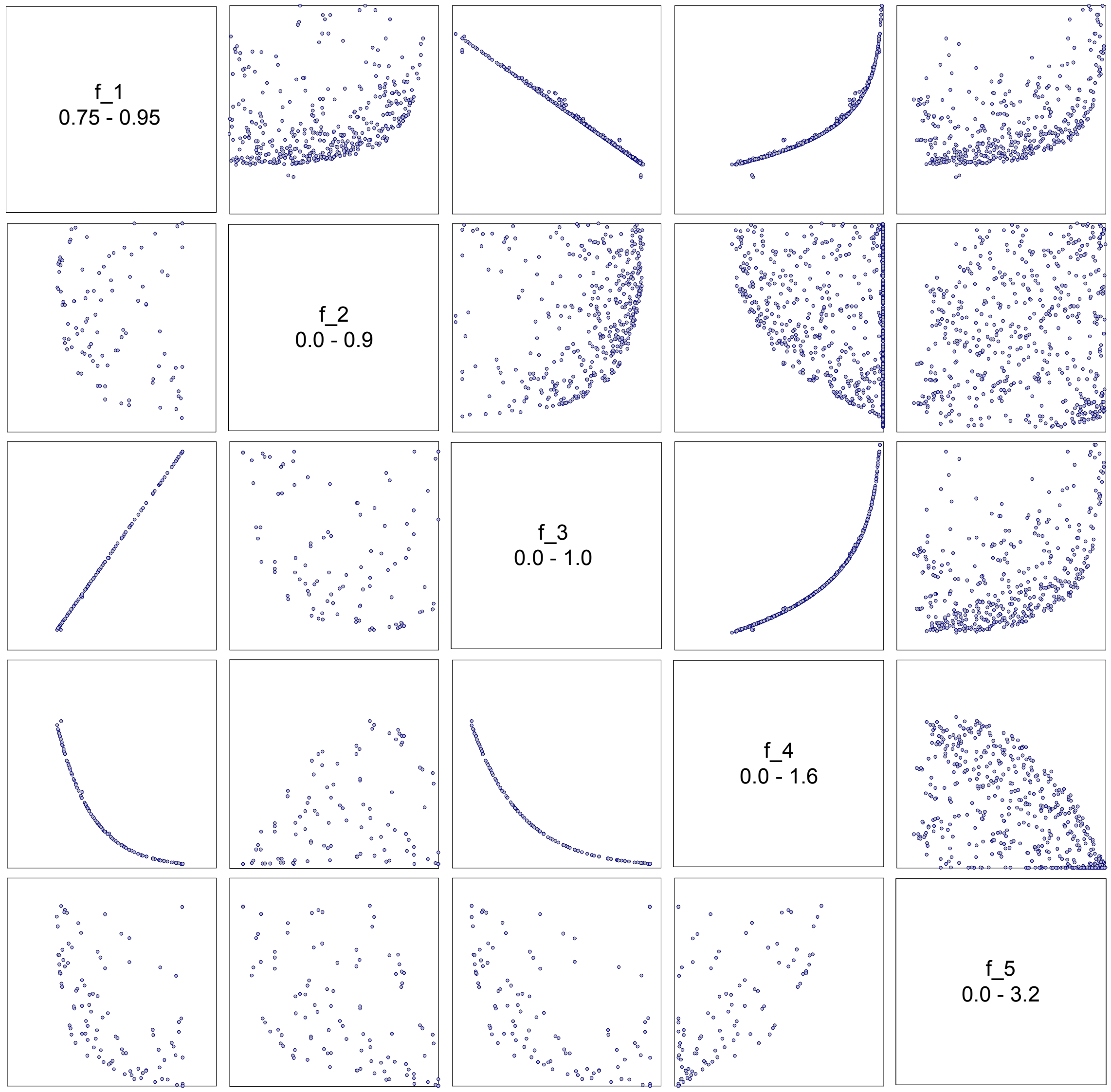


Figure 50: Non-dominated solutions on WATER with upper diagonal plots for ANSGA-II (adaptable N, p_m, η_m, and fixed p_c, η_c) and lower diagonal plots for NSGA-II with fixed parameter settings

Results of ANSGA-II with Adaptable p_c, p_m, η_c, η_m, and *Fixed N*

This variant of ANSGA-II supports self-adaptive crossover probability (p_c), self-adaptive crossover distribution index (η_c), self-adaptive mutation probability (p_m), and self-adaptive mutation distribution index (η_m). The population size is set to the same fixed parameter value used in the NSGA-II: N = 100. In addition, the number of generations is set to the same values used in the NSGA-II: G = 250 for SCH, FON, POL, KUR, ZDT1, ZDT2, ZDT3, ZDT4, ZDT6; and G = 500 for DEB, SRN TNK, WATER. Table 28 presents performance results of the ANSGA-II with adaptable p_c, p_m, η_c, η_m against the original NSGA-II with fixed parameter settings on thirteen benchmark problems.

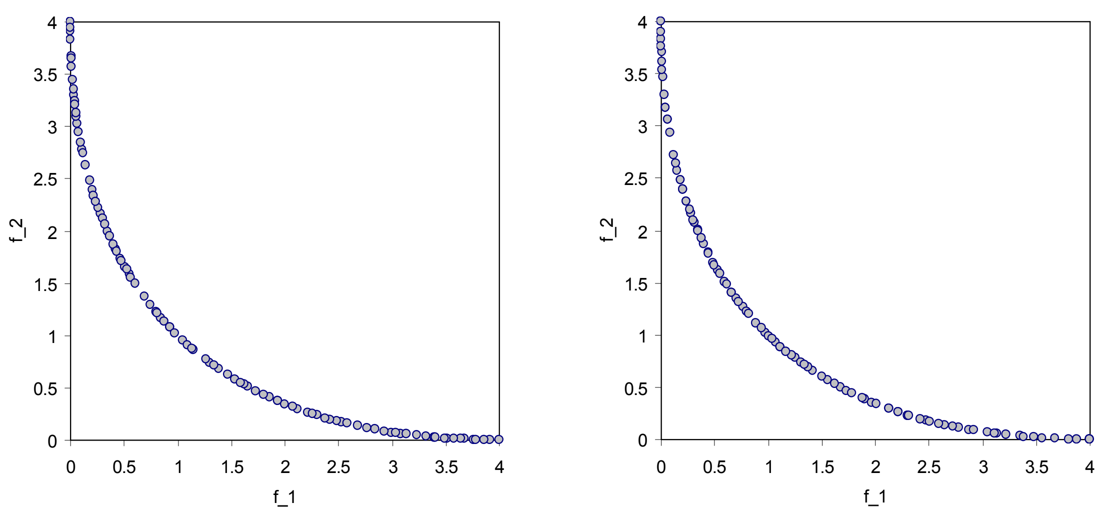
Table 28: Performance results of ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) against the original NSGA-II with fixed parameter settings

Problems	ANSGA-II						NSGA-II					
	N	G	Diversity Metric	Func. Eval.	Time (sec)	$N*G$ (1000)	N	G	Diversity Metric	Func. Eval.	Time (sec)	$N*G$ (1000)
SCH	100	250	0.5560	251	4	25	100	250	0.5711	251	4	25
FON	100	250	0.7370	251	5	25	100	250	0.7219	251	4	25
POL	100	250	0.9430	251	5	25	100	250	0.9538	251	3	25
KUR	100	250	0.8202	251	5	25	100	250	0.8121	251	4	25
ZDT1	100	250	0.7014	251	5	25	100	250	0.7431	251	4	25
ZDT2	100	250	0.7091	251	6	25	100	250	0.7390	251	6	25
ZDT3	100	250	0.7777	251	6	25	100	250	0.8731	251	5	25
ZDT4	100	250	0.7275	251	4	25	100	250	0.6766	251	5	25
ZDT6	100	250	0.7202	251	6	25	100	250	0.7046	251	8	25
DEB	100	500	0.7373	501	11	50	100	500	0.7772	501	15	50
SRN	100	500	0.7854	501	9	50	100	500	0.8011	501	16	50
TNK	100	500	0.8317	501	8	50	100	500	0.8072	501	8	50
WATER	100	500	0.7167	501	10	50	100	500	0.6274	501	10	50

For problems SCH, POL, ZDT1, ZDT2, ZDT3, DEB, and SRN the ANSGA-II with adaptable p_c, p_m, η_c, η_m out-performs the original NSGA-II with fixed parameter settings in terms of finding better spread of non-dominated solutions and approximating to the true Pareto-optimal front. This demonstrates that the ANSGA-II is able to learn good parameter values for p_c, p_m, η_c, η_m and the fixed settings (N = 100, G = 250) are good for the above problems. For two problems FON and TNK, the ANSGA-II with adaptable p_c, p_m, η_c, η_m performs very close to the NSGA-II with fixed parameter settings. For other problems (KUR, ZDT4, ZDT6, and WATER), this variant of the

ANSGA-II performs worse than the NSGA-II with fixed parameter settings. The most likely reason is that time spent on finding good parameter values for p_c, p_m, η_c, η_m is time taken away from finding diverse sets of non-dominated solutions. This variant has execution time comparable to that of required by the NSGA-II. This implies that the ANSGA-II is able to learn good values for crossover and mutation parameters quickly. This variant has shorter execution time than the variants with adaptable population size. This implies that the overhead for adapting population size is expensive. The plots of non-dominated solutions on thirteen benchmark problems obtained by the ANSGA-II and the NSGA-II are presented and compared in the following.

Results for the Two-Objective Test Problem SCH



(a) (b)

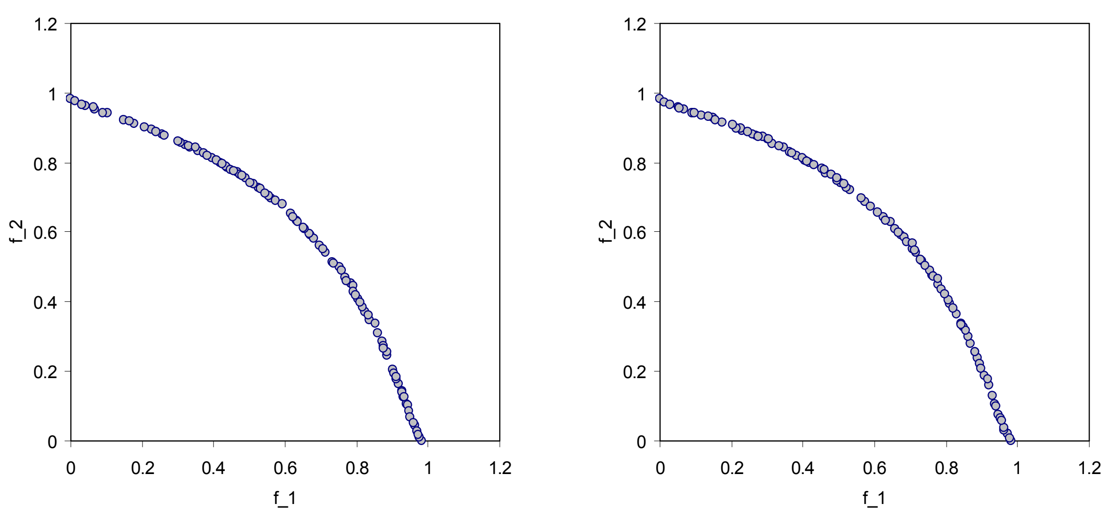
Figure 51: Non-dominated solutions on SCH with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.6968, average η_c = 768.65, average p_m = 0.9452, and average η_m = 1862.68. Figure 51 and diversity metric values for SCH in Table 28 show that the

ANSGA-II with adaptable p_c, p_m, η_c, η_m finds a better distribution of Pareto-optimal solutions (distribution of solutions is a little bit more uniform in Figure 51(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem SCH than the NSGA-II with fixed parameter settings.

Results for the Two-Objective Test Problem FON

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average $p_c = 0.8421$, average $\eta_c = 768.72$, average $p_m = 0.6827$, and average $\eta_m = 324.52$. Figure 52 and diversity metric values for FON in Table 28 show that the ANSGA-II with adaptable p_c, p_m, η_c, η_m finds a little bit less spread of Pareto-optimal solutions (more small gaps on the Pareto-optimal front of Figure 52(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) in solving the problem FON than the NSGA-II with fixed parameter settings. However, the distribution of Pareto-optimal solutions with ANSGA-II is adequate ($dm_{\text{ANSGA-II}} \leq 0.80$).



(a) (b)

Figure 52: Non-dominated solutions on FON with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem POL

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.6549, average η_c = 188.04, average p_m = 0.9543, and average η_m = 253.15. Figure 53 and diversity metric values for POL in Table 28 show that the ANSGA-II with adaptable p_c, p_m, η_c, η_m finds a better distribution of Pareto-optimal solutions (distribution of solutions is more continuous on the bottom Pareto-optimal front of Figure 53(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem POL than the NSGA-II with fixed parameter settings.

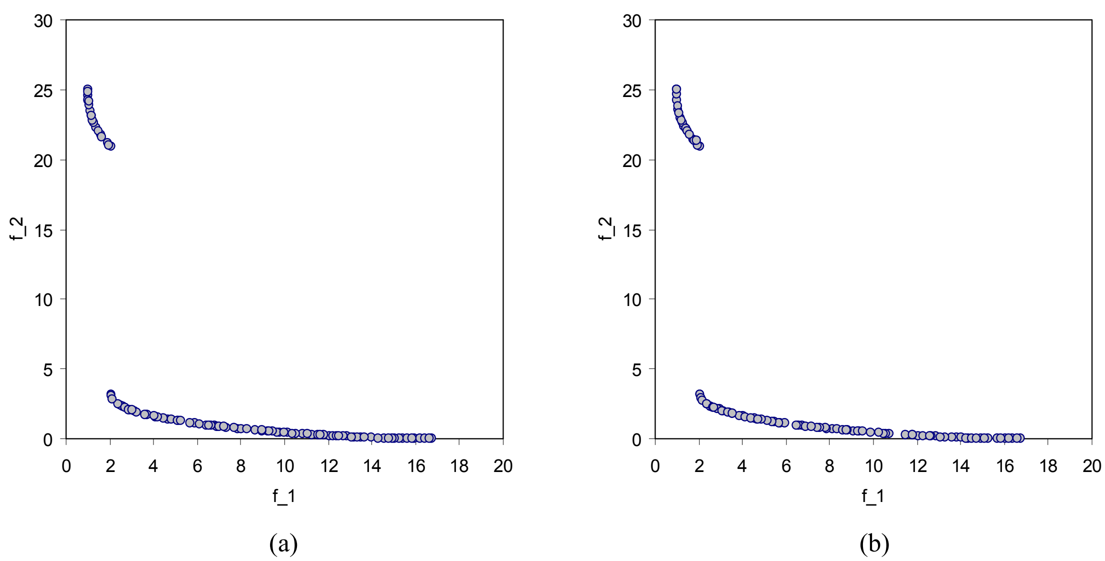


Figure 53: Non-dominated solutions on POL with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem KUR

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.6325, average η_c = 698.68, average p_m = 0.7934, and average η_m = 484.96. Figure 54(a) shows that the ANSGA-II with adaptable p_c, p_m, η_c, η_m

performs worse than the NSGA-II because it fails to obtain non-dominated solutions that cover the entire shape of the Pareto-optimal front (bottom right region).

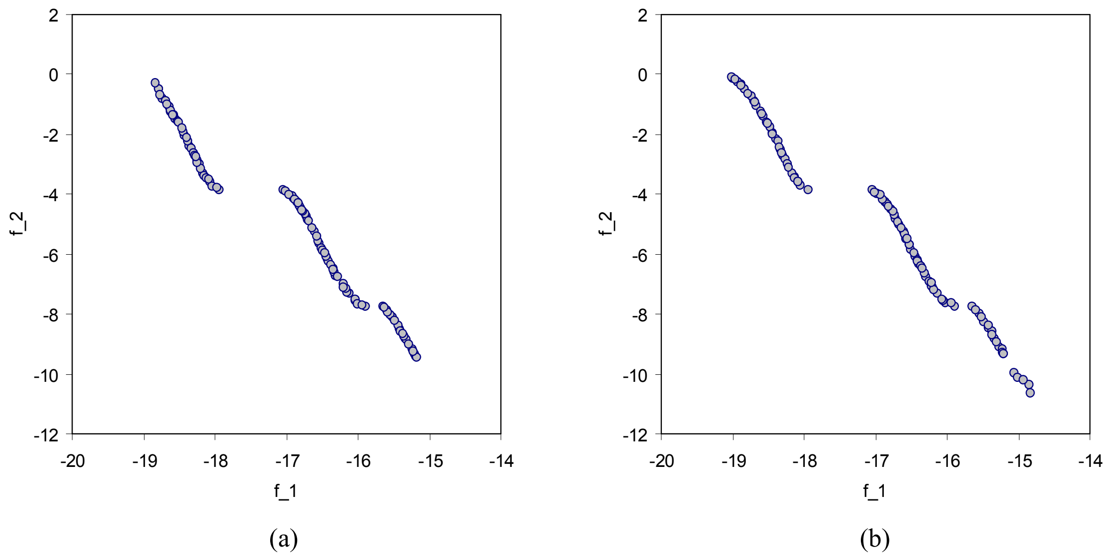


Figure 54: Non-dominated solutions on KUR with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT1

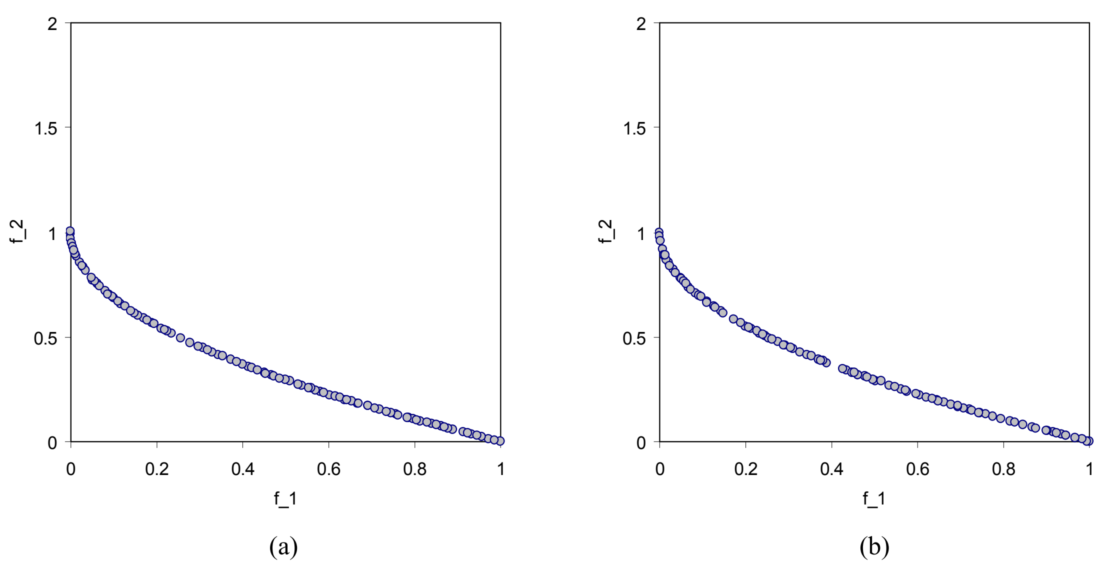
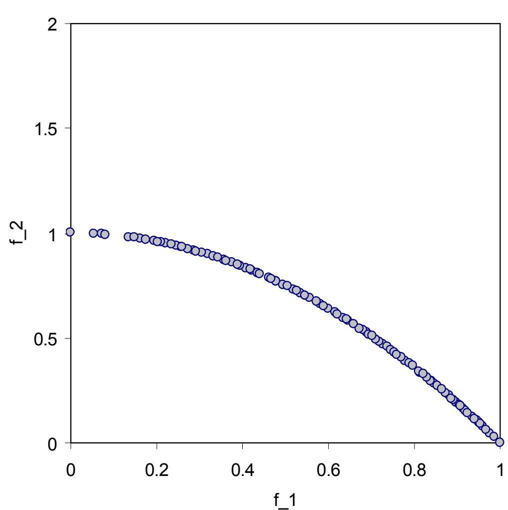


Figure 55: Non-dominated solutions on ZDT1 with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

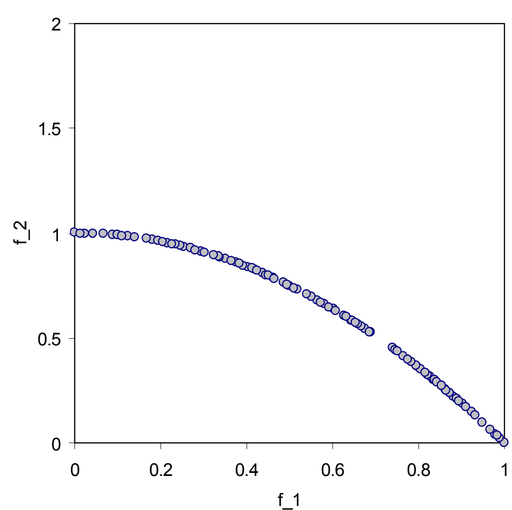
The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.0478, average η_c = 49.85, average p_m = 0.8651, and average η_m = 53.63. Figure 55 and diversity metric values for ZDT1 in Table 28 show that the ANSGA-II with adaptable p_c, p_m, η_c, η_m finds a better distribution of Pareto-optimal solutions (less gaps on the Pareto-optimal front of Figure 55(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem ZDT1 than the NSGA-II with fixed parameter settings.

Results for the Two-Objective Test Problem ZDT2

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.3757, average η_c = 25.61, average p_m = 0.8744, and average η_m = 32.22. Figure 56 and diversity metric values for ZDT2 in Table 28 show that the ANSGA-II with adaptable p_c, p_m, η_c, η_m finds a better distribution of Pareto-optimal solutions (less gaps on the Pareto-optimal front of Figure 56(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem ZDT2 than the NSGA-II with fixed parameter settings.



(a)

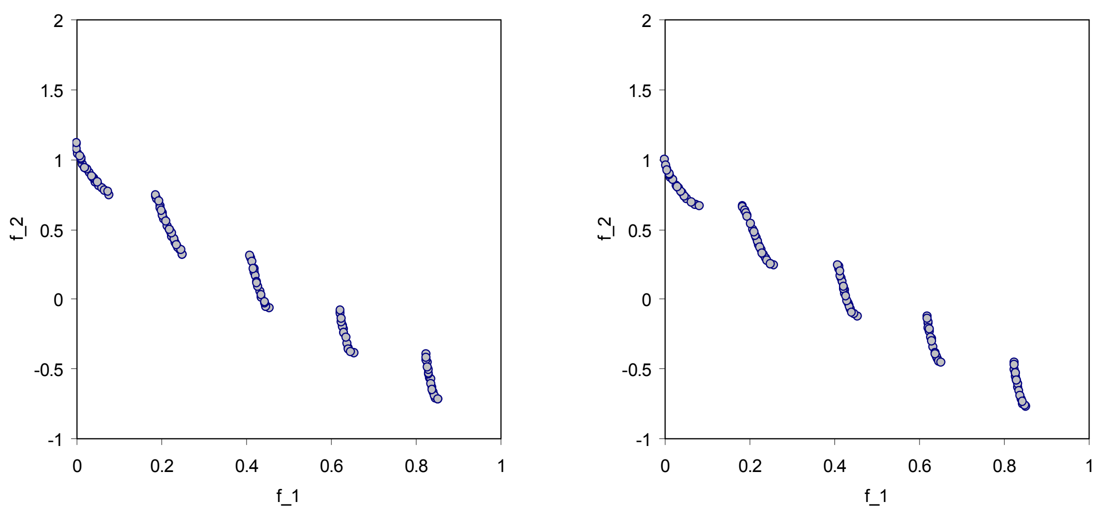


(b)

Figure 56: Non-dominated solutions on ZDT2 with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT3

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.2836, average η_c = 275.43, average p_m = 0.1725, and average η_m = 301.43. Figure 57 and diversity metric values for ZDT3 in Table 28 show that the ANSGA-II with adaptable p_c, p_m, η_c, η_m finds a better distribution of Pareto-optimal solutions (distribution of solutions is about the same visually in Figure 57(a) and Figure 57(b) but $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem ZDT3 than the NSGA-II with fixed parameter settings.

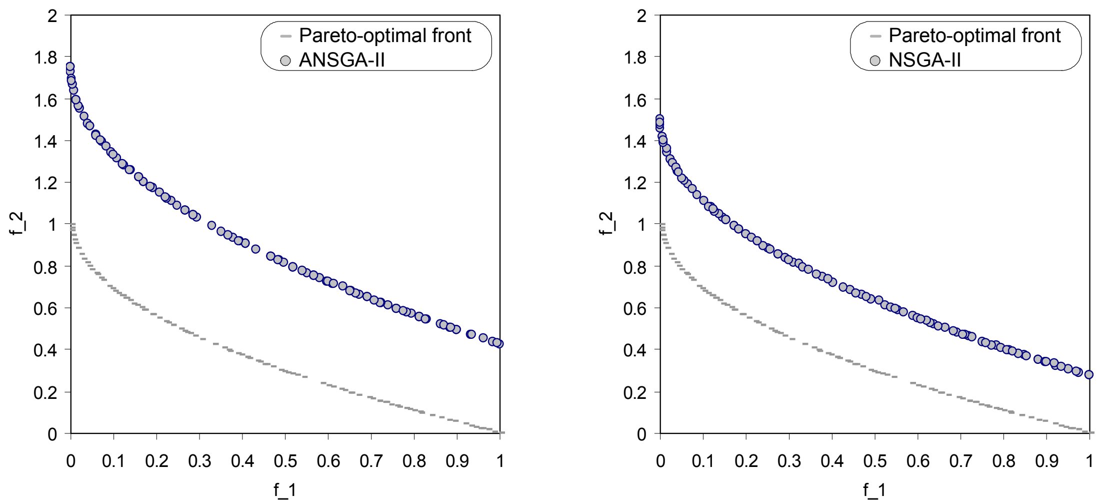


(a) (b)

Figure 57: Non-dominated solutions on ZDT3 with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT4

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.8428, average η_c = 125.19, average p_m = 0.1103, and average η_m = 132.54. Figure 58 shows that both the ANSGA-II with adaptable p_c, p_m, η_c, η_m and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT4. However, the NSGA-II approximates closer to the global Pareto-optimal front. This implies that given the fixed population size N = 100 and number of generations G = 250, the ANSGA-II does not have enough time to learn good parameter values for p_c, p_m, η_c, η_m for the problem ZDT4.

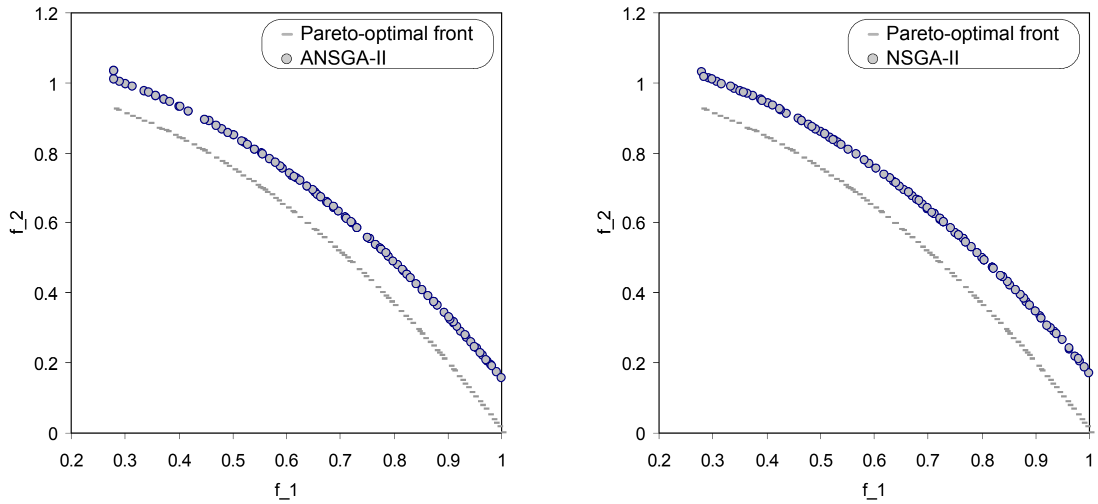


(a) (b)

Figure 58: Non-dominated solutions on ZDT4 with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT6

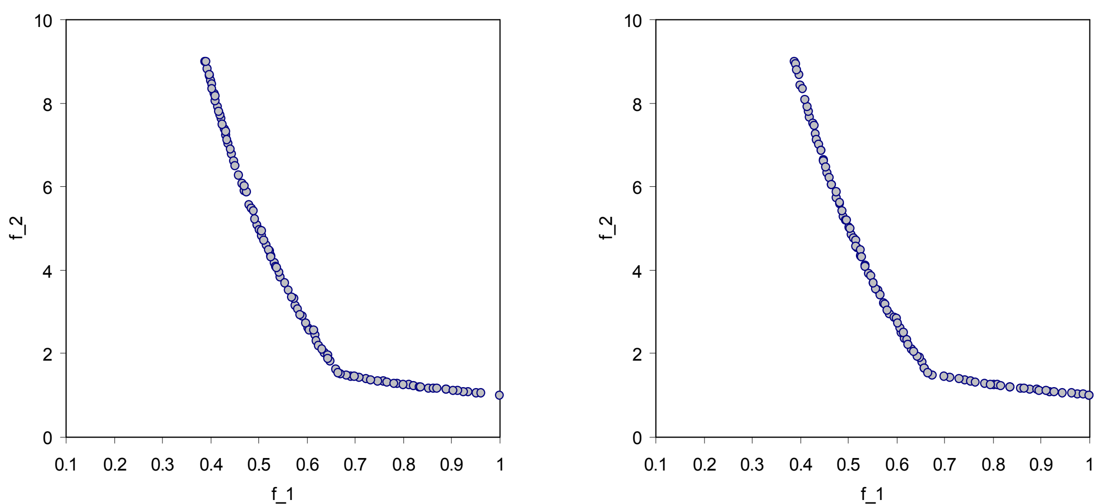
The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.0612, average η_c = 40.30, average p_m = 0.3142, and average η_m = 264.83. Similar to the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m, Figure 59 shows that both ANSGA-II with adaptable p_c, p_m, η_c, η_m and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT6. However, this variant generates one non-dominated solution that does not line up on the non-dominated front and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$.



(a) (b)

Figure 59: Non-dominated solutions on ZDT6 with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem DEB



(a) (b)

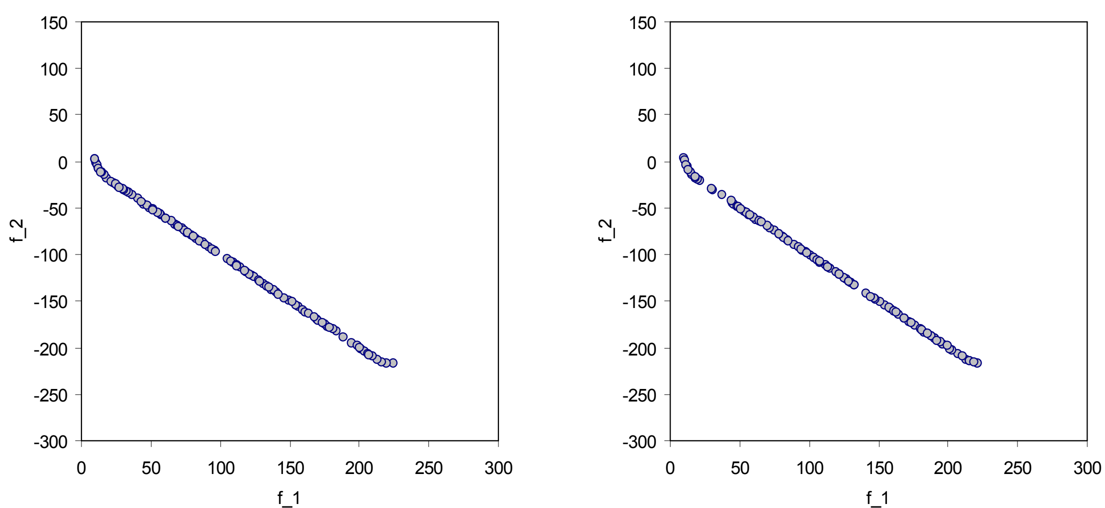
Figure 60: Non-dominated solutions on DEB with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.2136, average η_c = 590.48, average p_m = 0.9490, and average η_m = 184.91. Figure 60 and diversity metric values for DEB in Table 28 show that the

ANSGA-II with adaptable p_c, p_m, η_c, η_m finds a little bit better spread of Pareto-optimal solutions (distribution of solutions is about the same visually in Figure 60(a) and Figure 60(b) but $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) with less time in solving the problem DEB than the NSGA-II with fixed parameter settings.

Results for the Two-Objective Test Problem SRN

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average $p_c = 0.9511$, average $\eta_c = 175.08$, average $p_m = 0.7373$, and average $\eta_m = 150.12$. Figure 61 and diversity metric values for SRN in Table 28 show that the ANSGA-II with adaptable p_c, p_m, η_c, η_m finds a better spread of Pareto-optimal solutions (less small gaps on the Pareto-optimal front of Figure 61(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} -$ 0.001) with less time in solving the problem SRN than the NSGA-II with fixed parameter settings.



(a) (b)

Figure 61: Non-dominated solutions on SRN with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem TNK

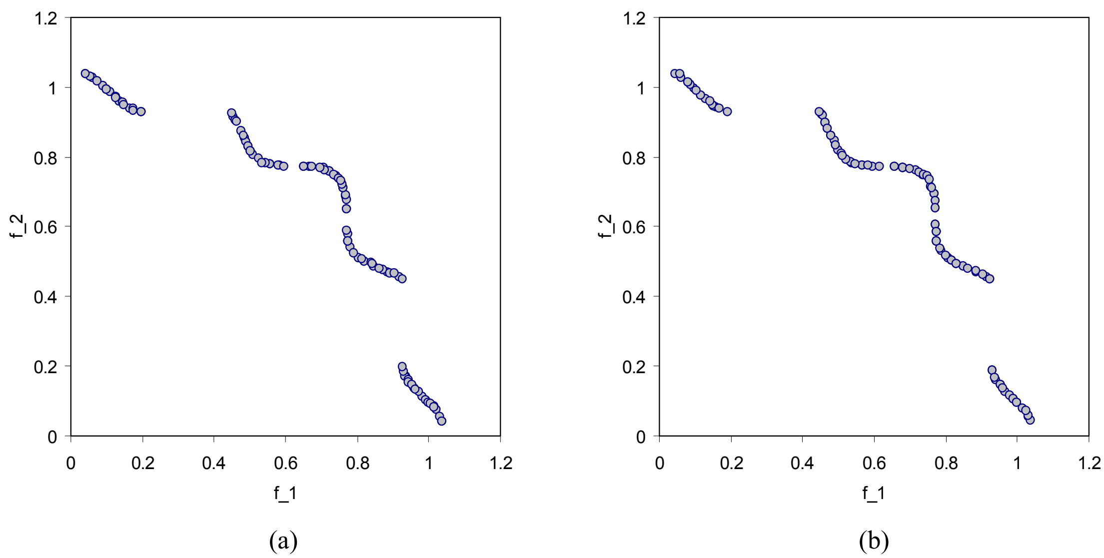


Figure 62: Non-dominated solutions on TNK with (a) ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.2823, average η_c = 586.14, average p_m = 0.6118, and average η_m = 433.32. Figure 62 and diversity metric values for TNK in Table 28 show that the ANSGA-II with adaptable p_c, p_m, η_c, η_m finds a little bit less spread of Pareto-optimal solutions (gaps are little bit wider on the middle Pareto-optimal front of Figure 62(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) in solving the problem TNK than the NSGA-II with fixed parameter settings. The diversity metric values (0.8317 for ANSGA-II and 0.8272 for NSGA-II) are little bit higher than the adequate diversity metric value (0.80) because the way the diversity metric calculation handling discontinuous Pareto-optimal fronts. However, observing visually the plots in Figure 62(a), it can be determined that the distribution of Pareto-optimal solutions with ANSGA-II is adequate.

Results for the Five-Objective Real-World Problem WATER

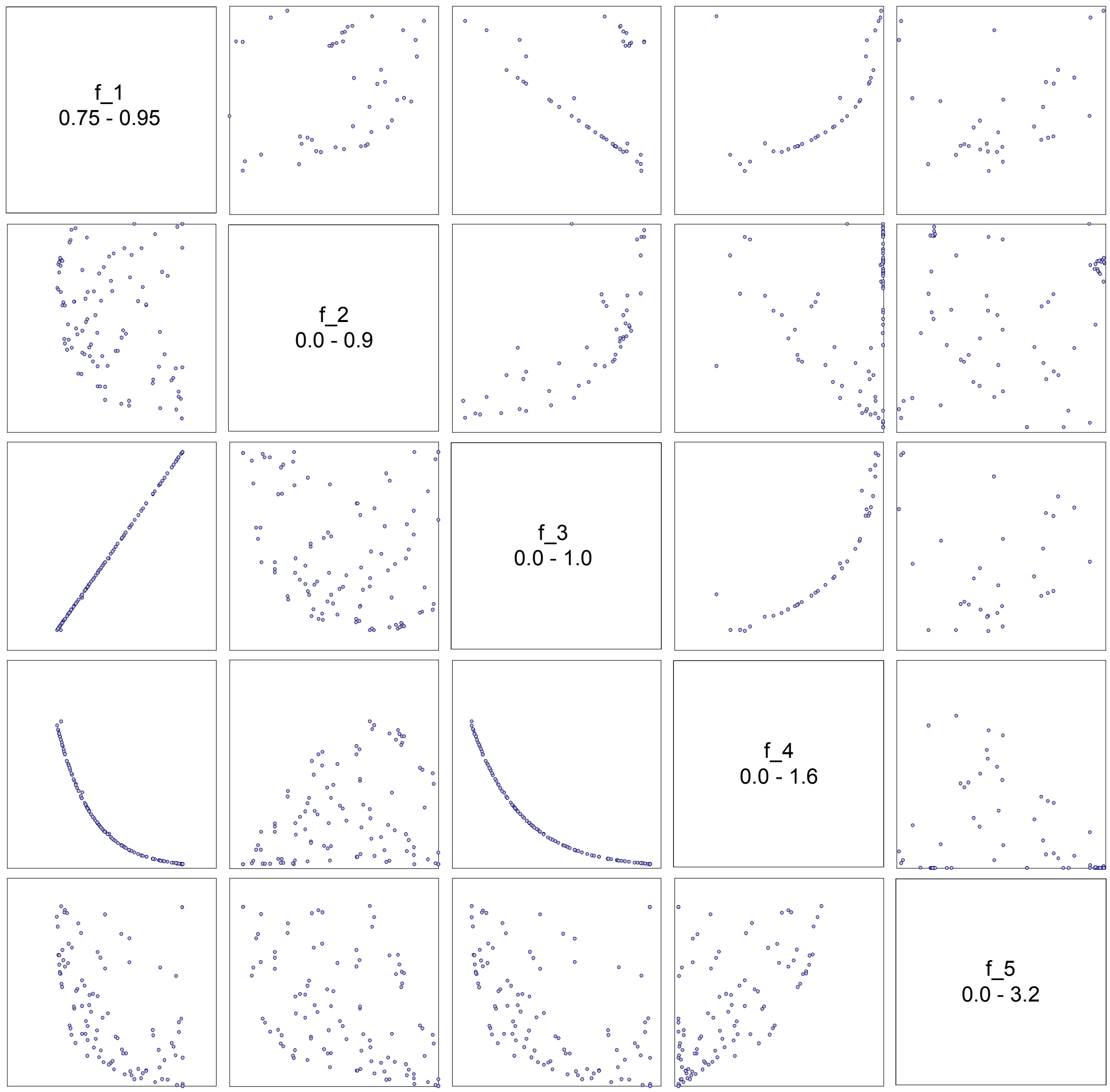


Figure 63: Non-dominated solutions on WATER with upper diagonal plots for ANSGA-II (adaptable p_c, p_m, η_c, η_m, and fixed N) and lower diagonal plots for NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.2639, average η_c = 514.22, average p_m = 0.8207, and average η_m = 193.28. Figure 63 and diversity metric values for WATER in Table 28 show that the ANSGA-II with adaptable p_c, p_m, η_c, η_m finds much less spread of non-dominated solutions (less distribution of solutions on the Pareto-optimal fronts in upper diagonal plots of Figure 63(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) in solving the problem WATER than the NSGA-II with fixed parameter settings. The plots for ANSGA-II have less formed patterns than the plots for NSGA-II, implying that this variant of ANSGA-II achieves less convergence than NSGA-II on this complex problem.

Results of ANSGA-II with Adaptable N, and Fixed p_c, p_m, η_c, η_m

This variant of ANSGA-II supports adaptive population size (N). The crossover probability (p_c), crossover distribution index (η_c), mutation probability (p_m), and mutation distribution index (η_m) are set to the same parameter values used in the NSGA-II: p_c = 0.9, η_c = 20, p_m = 0.5, and η_m = 100. Table 29 presents performance results of the ANSGA-II with adaptable N against the original NSGA-II with fixed parameter settings on thirteen benchmark problems.

Table 29: Performance results of ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) against the original NSGA-II with fixed parameter settings

Problems	ANSGA-II						NSGA-II					
	N	G	Diversity Metric	Func. Eval.	Time (sec)	$N*G$ (1000)	N	G	Diversity Metric	Func. Eval.	Time (sec)	$N*G$ (1000)
SCH	40	286	0.6321	554	12	11.44	100	250	0.5711	251	4	25
FON	40	252	0.7173	689	15	10.08	100	250	0.7219	251	4	25
POL	80	564	0.9462	1183	63	45.12	100	250	0.9538	251	3	25
KUR	160	352	0.7942	899	64	56.32	100	250	0.8121	251	4	25
ZDT1	40	292	0.7524	620	17	11.68	100	250	0.7431	251	4	25
ZDT2	80	368	0.7338	926	38	29.44	100	250	0.7390	251	6	25
ZDT3	40	288	0.7402	709	15	11.52	100	250	0.8731	251	5	25
ZDT4	40	314	0.4736	675	14	12.56	100	250	0.6766	251	5	25
ZDT6	40	328	0.7532	777	16	13.12	100	250	0.7046	251	8	25
DEB	80	368	0.7840	812	28	29.44	100	500	0.7772	501	15	50
SRN	40	270	0.7787	494	13	10.8	100	500	0.8011	501	16	50
TNK	640	2592	0.8057	5904	1468	1658.88	100	500	0.8072	501	8	50
WATER	200	526	0.6771	1145	346	105.2	100	500	0.6274	501	10	50

For problems FON, POL, KUR, ZDT2, and SRN, the ANSGA-II with adaptable N out-performs the original NSGA-II with fixed parameter settings in terms of finding better distribution of non-dominated solutions and approximating to the true Pareto-optimal front. This demonstrates that this variant of the ANSGA-II is able to learn good parameter value for N and the fixed settings (p_c = 0.9, p_m = 0.5, η_c = 20, η_m = 100) are good for these problems. For problems SCH, ZDT1, and DEB, the ANSGA-II with adaptable N performs very close to the NSGA-II with fixed parameter settings. The results on the problem DEB are almost the same in both algorithms. Both this variant of ANSGA-II and NSGA-II fail to converge to the global Pareto-optimal front on the problem ZDT6. For other problems (ZDT3, ZDT4, TNK and WATER), ANSGA-II with adaptable N performs worse than the NSGA-II with fixed parameter settings. The most likely reason is that there is not enough diversity of solutions in the population but the distribution of obtained solutions is good; therefore, the ANSGA-II terminates prematurely. This variant of the ANSGA-II requires less time to solve the problem SRN than the NSGA-II. For other problems, this variant is slower than the NSGA-II due to overheads, which appear to be acceptable (except for problem TNK), of solving the problem and learning good parameter value at the same time. The plots of non-dominated solutions on thirteen benchmark problems obtained by the ANSGA-II and the NSGA-II are presented and compared in the following.

Results for the Two-Objective Test Problem SCH

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 40. Figure 64 and diversity metric values for SCH in Table 29 show that the ANSGA-II with adaptable N finds a little bit less distribution of Pareto-optimal solutions (distribution of solutions is less uniform in Figure 64(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) using a much smaller population size in solving the problem SCH than the NSGA-II with fixed parameter settings. However, the distribution of Pareto-optimal solutions with ANSGA-II is adequate ($dm_{\text{ANSGA-II}} \leq 0.80$).

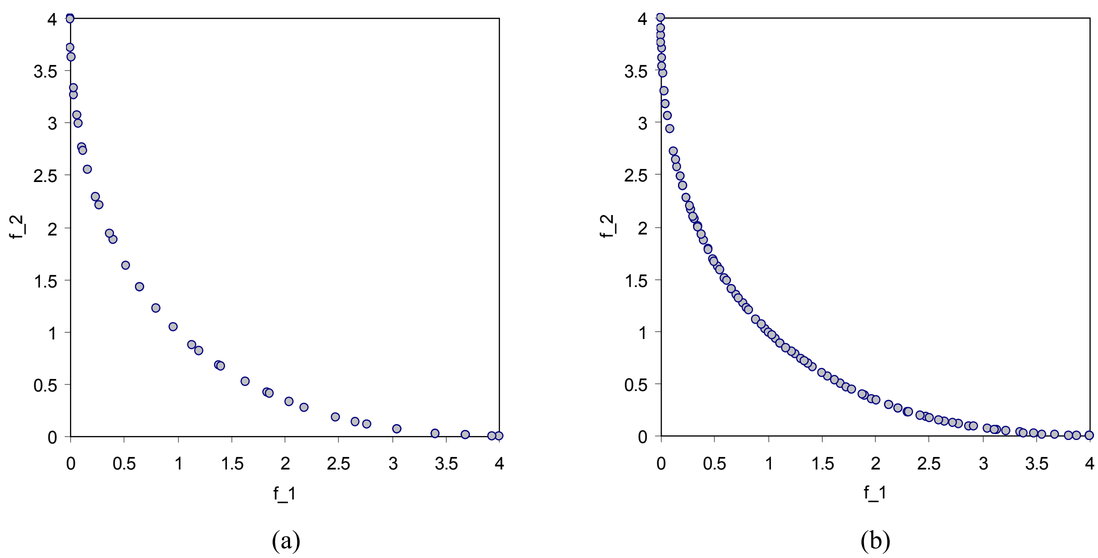
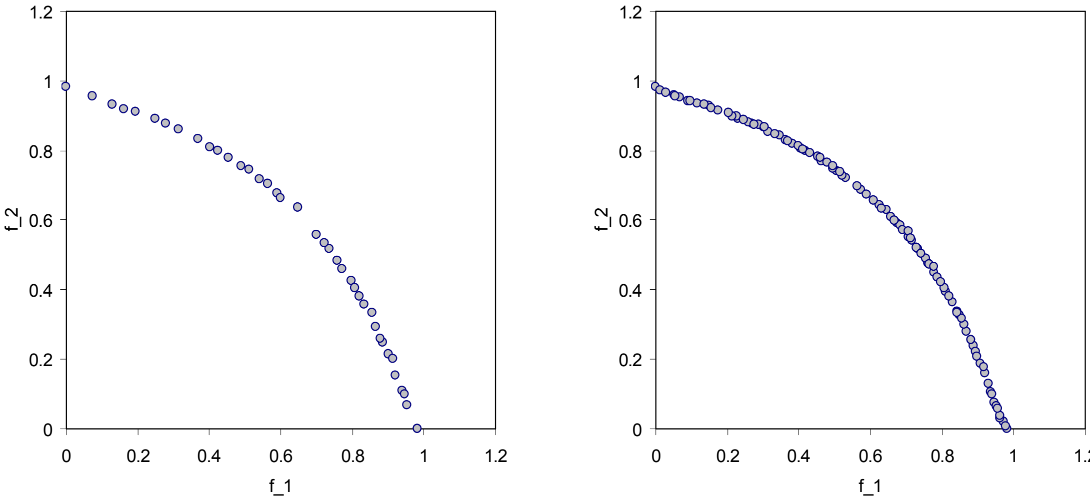


Figure 64: Non-dominated solutions on SCH with (a) ANSGA-II (adaptable *N*, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem FON

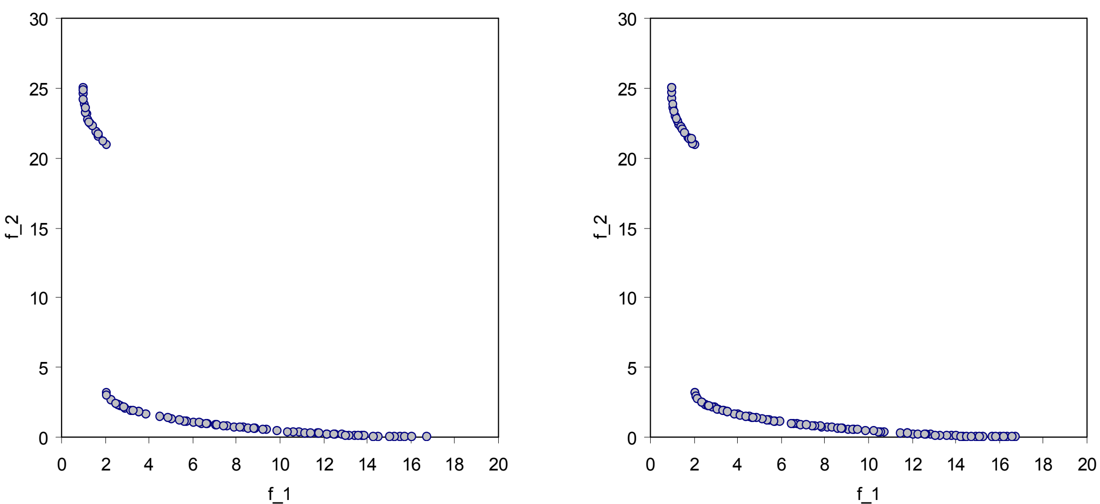
The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 40. Figure 65 and diversity metric values for FON in Table 29 show that the ANSGA-II with adaptable N finds a wide and uniformly spread of Pareto-optimal solutions (less crowded solutions on the Pareto-optimal front of Figure 65(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a much smaller population size in solving the problem FON than the NSGA-II with fixed parameter settings.



(a) (b)

Figure 65: Non-dominated solutions on FON with (a) ANSGA-II (adaptable *N*, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem POL



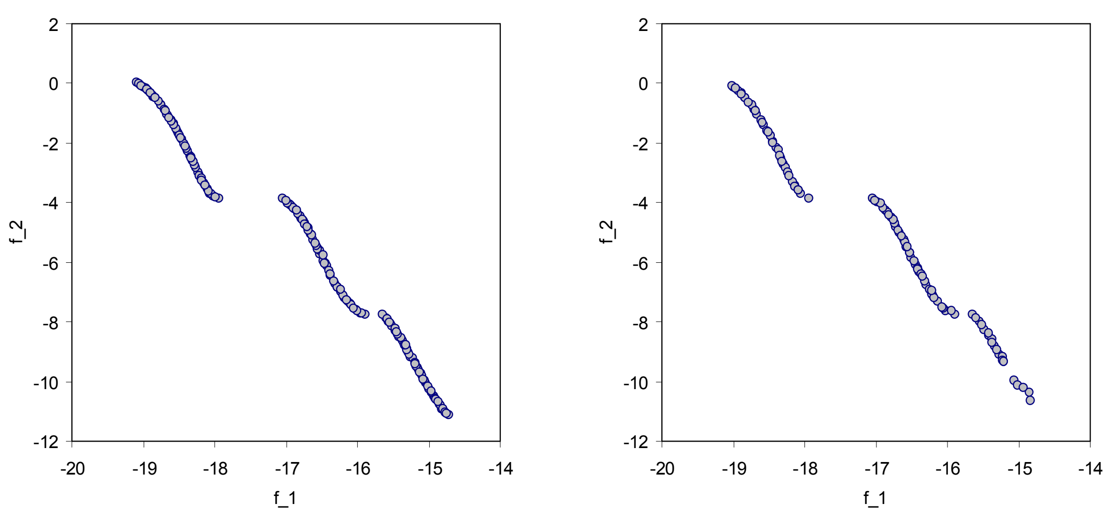
(a) (b)

Figure 66: Non-dominated solutions on POL with (a) ANSGA-II (adaptable *N*, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 80. Figure 66 and diversity metric values for POL in Table 29 show that the ANSGA-II with adaptable N finds a better spread of Pareto-optimal solutions (distribution of solutions is less crowded and more uniform in Figure 66(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem POL than the NSGA-II with fixed parameter settings. The algorithm requires a smaller population size but a larger number of generations to solve this problem than the NSGA-II. As a result, the number of solutions evaluated to find the non-dominated set is higher than that of required by the NSGA-II.

Results for the Two-Objective Test Problem KUR

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 160. Figure 67 and diversity metric values for KUR in Table 29 show that the ANSGA-II with adaptable N requires a larger population size to find a better spread of Pareto-optimal solutions (distribution of solutions is more continuous on the bottom Pareto-optimal front of Figure 67(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem KUR than the NSGA-II with fixed parameter settings.



(a) (b)

Figure 67: Non-dominated solutions on KUR with (a) ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT1

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 40. Figure 68 and diversity metric values for ZDT1 in Table 29 show that the ANSGA-II with adaptable N finds a little bit less spread of Pareto-optimal solutions (distribution of solutions is less uniform in Figure 68(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) using a much smaller population size in solving the problem ZDT1 than the NSGA-II with fixed parameter settings.

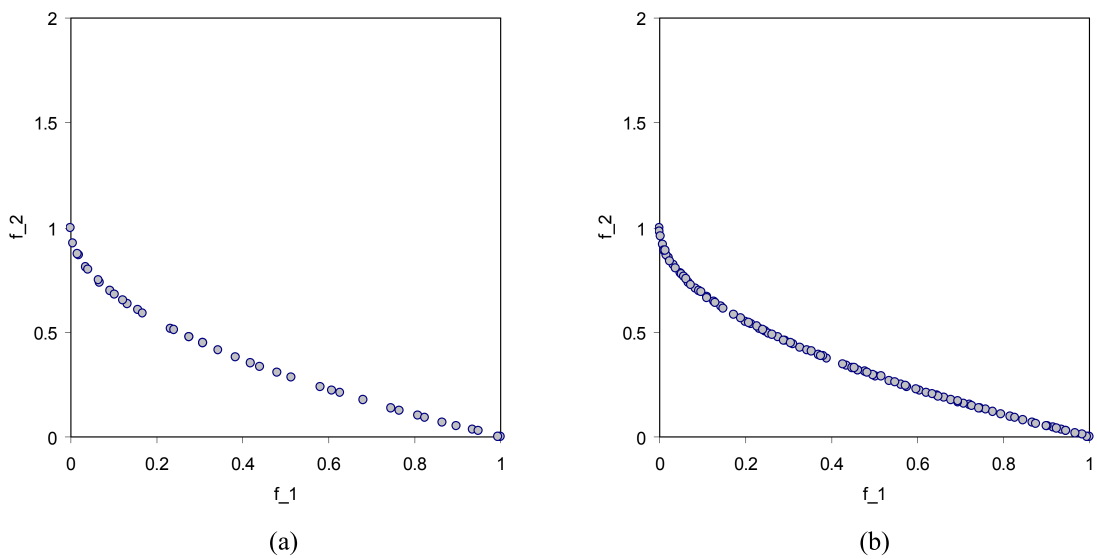


Figure 68: Non-dominated solutions on ZDT1 with (a) ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT2

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 80. Figure 69 and diversity metric values for ZDT2 in Table 29 show that the ANSGA-II with adaptable N finds a little bit better spread of Pareto-optimal solutions (distribution of solutions is less crowded and little bit more uniform in Figure 69(a) and

$dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a smaller population size in solving the problem ZDT2 than the NSGA-II with fixed parameter settings.

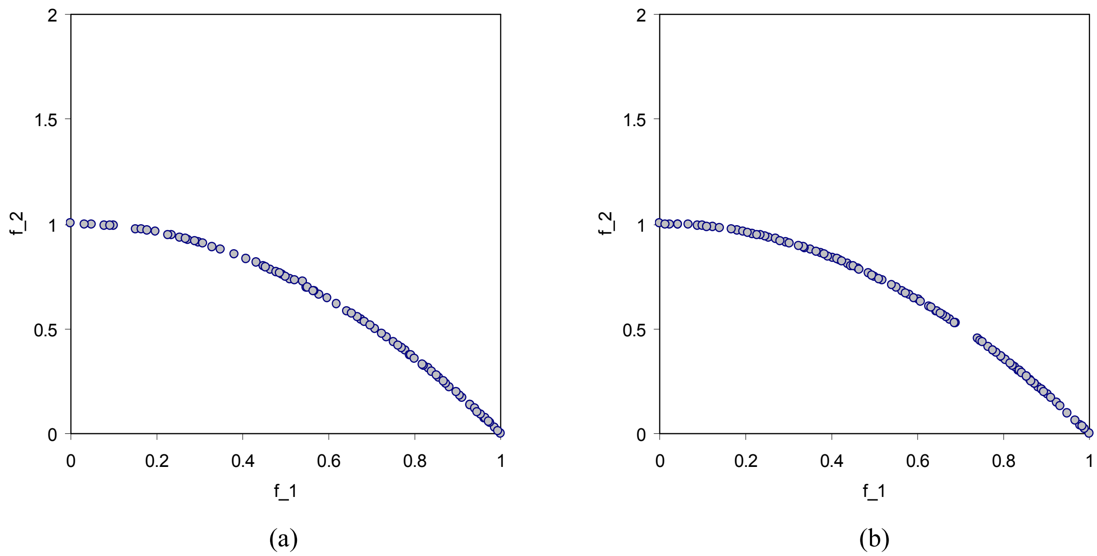


Figure 69: Non-dominated solutions on ZDT2 with (a) ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT3

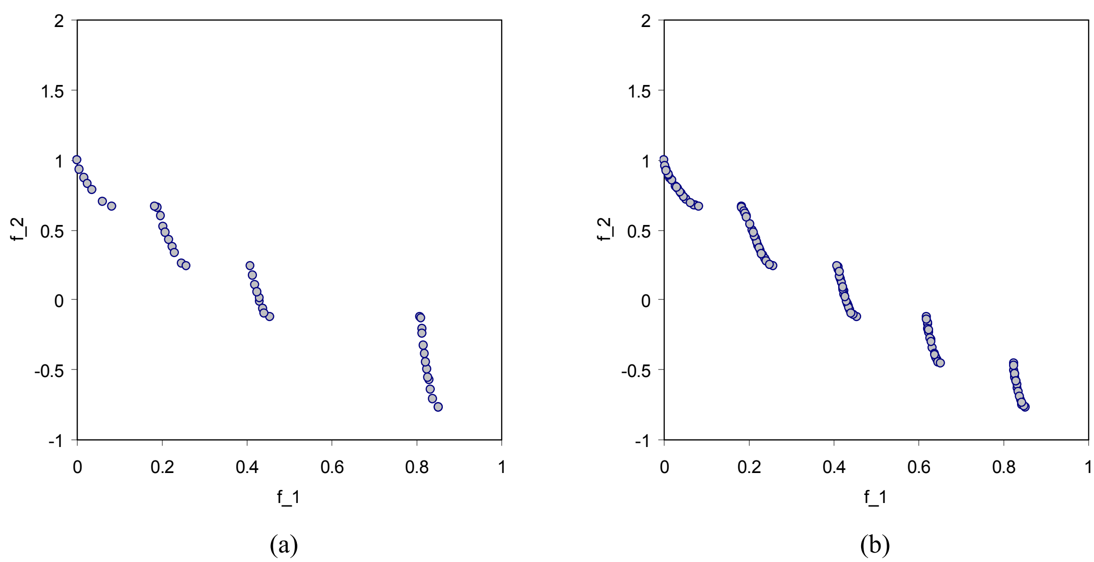
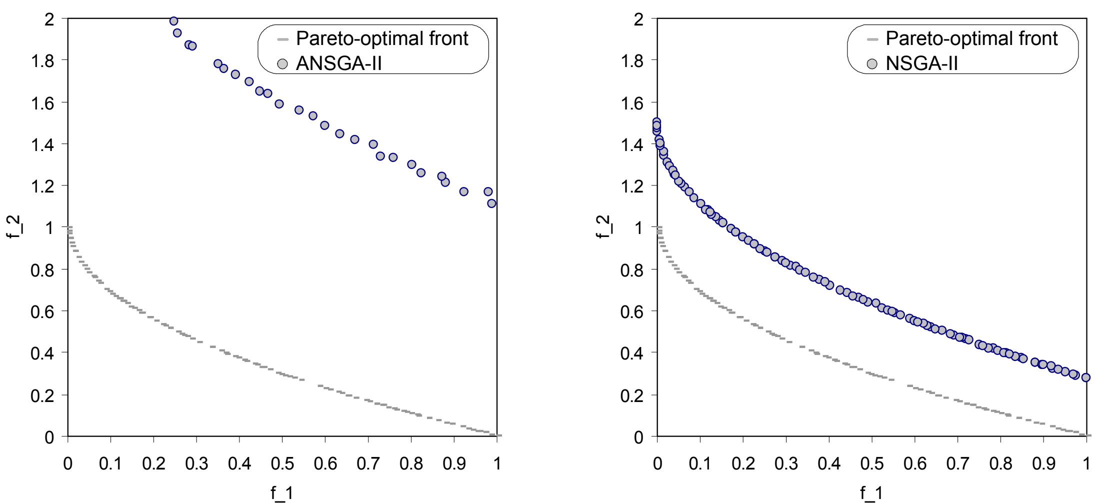


Figure 70: Non-dominated solutions on ZDT3 with (a) ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 40. Figure 70 shows that the ANSGA-II with adaptable N fails to find all discontinuous Pareto-optimal fronts for the problem ZDT3. It also fails to approximate the bottom non-dominated solution set to the true Pareto-optimal front.

Results for the Two-Objective Test Problem ZDT4

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 40. Figure 71 shows that both the ANSGA-II with adaptable N and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT4. However, the NSGA-II approximates closer to the global Pareto-optimal front.



(a) (b)

Figure 71: Non-dominated solutions on ZDT4 with (a) ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT6

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 40. Similar to the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m, Figure

72 shows that both ANSGA-II with adaptable N and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT6.

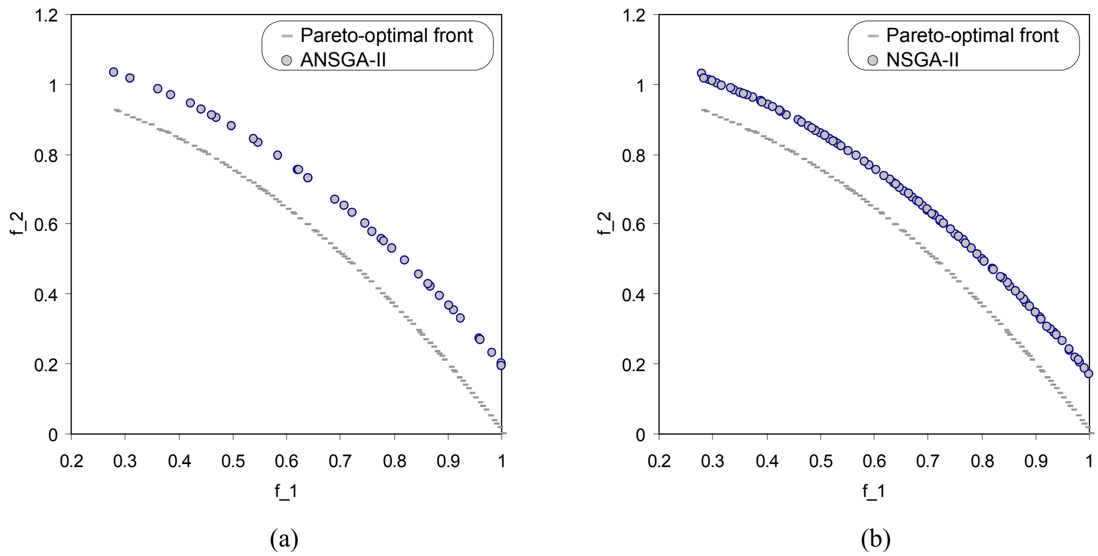


Figure 72: Non-dominated solutions on ZDT6 with (a) ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem DEB

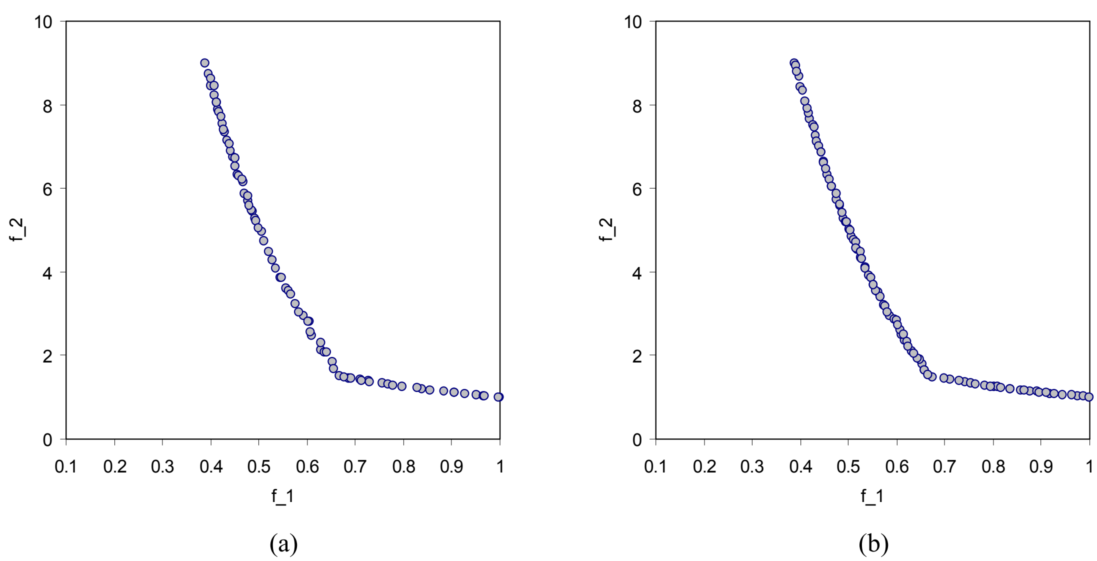
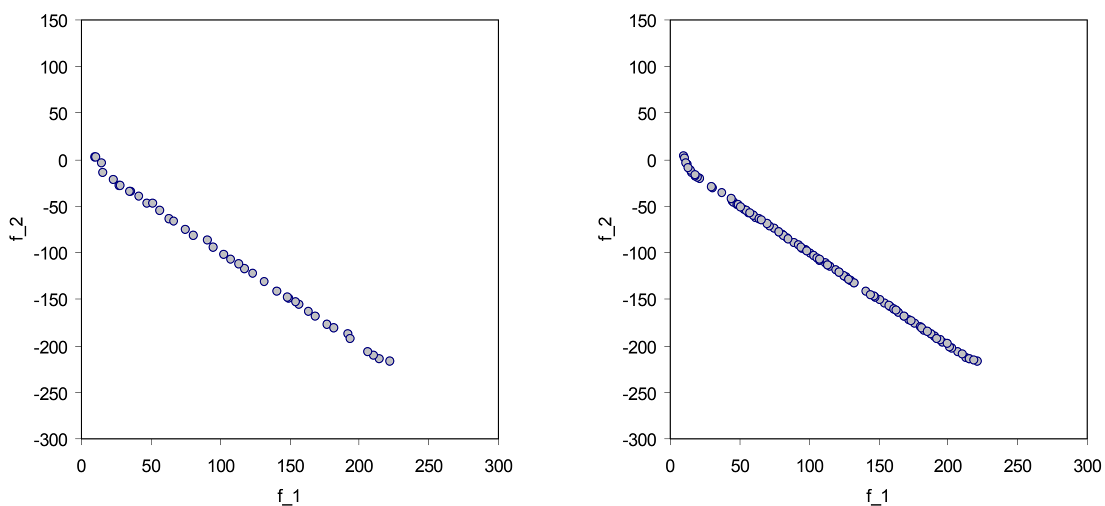


Figure 73: Non-dominated solutions on DEB with (a) ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 80. Figure 73 and diversity metric values for DEB in Table 29 show that the ANSGA-II with adaptable N finds a less spread of non-dominated solutions (distribution of solutions is a little bit less uniform in Figure 73(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) using a smaller population size in solving the problem DEB compared to the NSGA-II with fixed parameter settings. However, the distribution of Pareto-optimal solutions with ANSGA-II is adequate ($dm_{\text{ANSGA-II}} \leq 0.80$).

Results for the Two-Objective Test Problem SRN



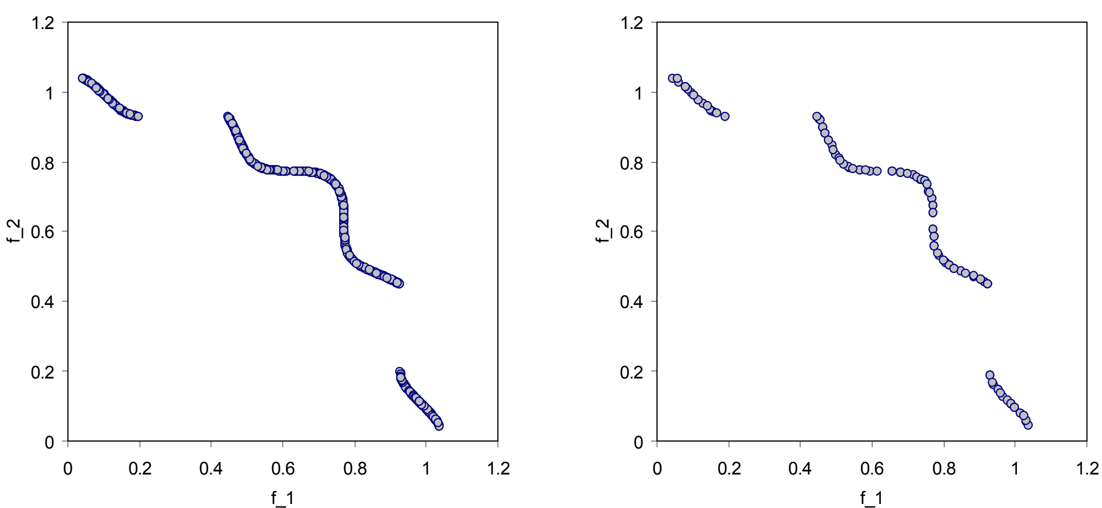
(a) (b)

Figure 74: Non-dominated solutions on SRN with (a) ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 40. Figure 74 and diversity metric values in Table 29 show that the ANSGA-II with adaptable N finds a little bit better spread of non-dominated solutions with good convergence (one solution does not line up on the front but distribution of solutions is less crowded in Figure 74(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) using a smaller population size in solving the problem SRN compared to the NSGA-II with fixed parameter settings. This variant takes less execution time to solve the problem SRN than the NSGA-II implying that the ANSGA-II can find a proper population size for this problem quickly.

Results for the Two-Objective Test Problem TNK

The value of the adaptable parameter identified by the ANSGA-II for this problem is: N = 640. Figure 75 and diversity metric values for TNK in Table 29 show that the ANSGA-II with adaptable N, p_m, η_m finds a better spread of Pareto-optimal solutions (distribution of solutions is more continuous in the middle Pareto-optimal front of Figure 75(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem TNK than the NSGA-II with fixed parameter settings. However, the ANSGA-II requires a much bigger population size and it takes an unreasonable long time to solve this problem.



(a) (b)

Figure 75: Non-dominated solutions on TNK with (a) ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Five-Objective Real-World Problem WATER

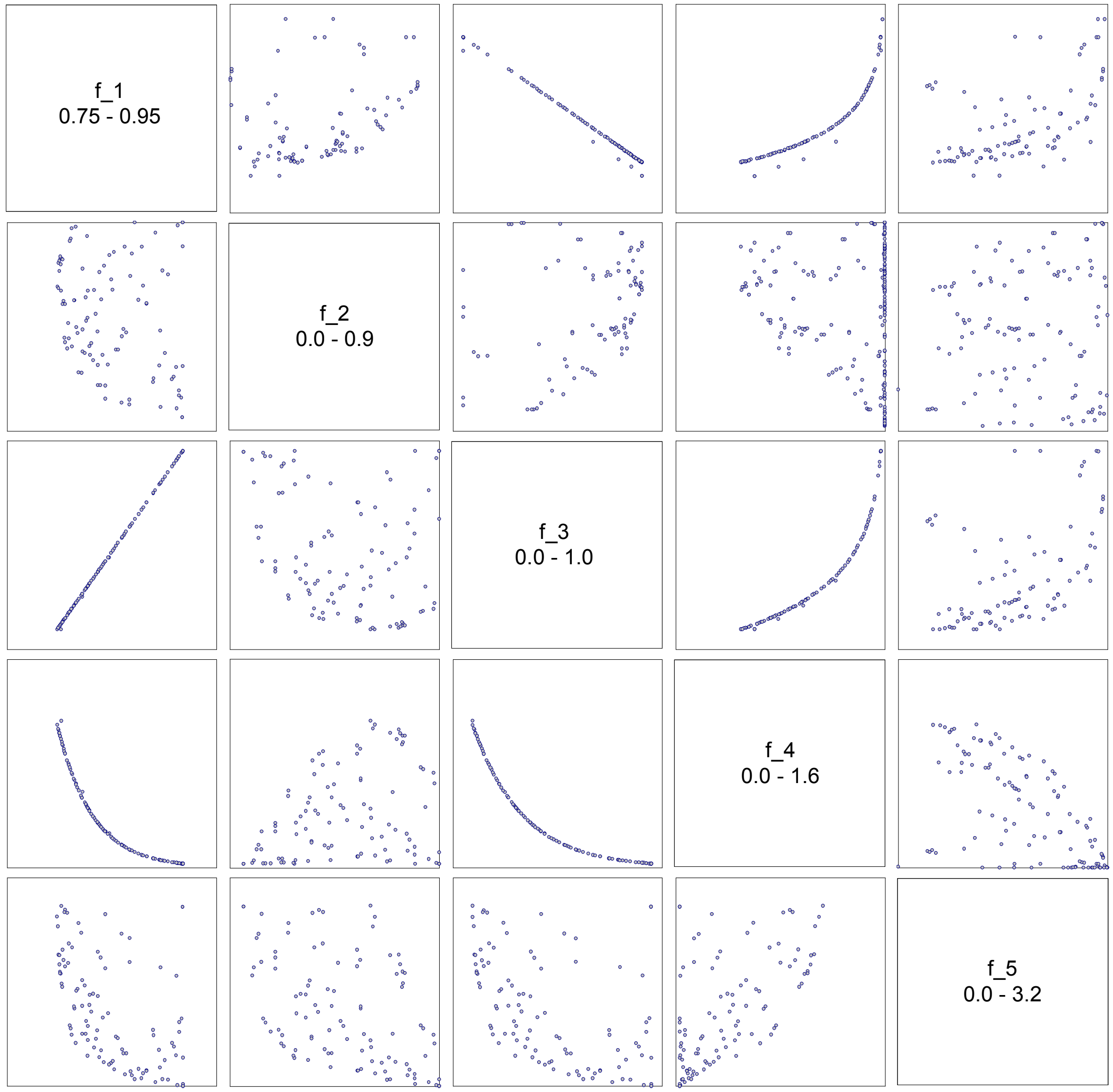


Figure 76: Non-dominated solutions on WATER with upper diagonal plots for ANSGA-II (adaptable N, and fixed p_c, p_m, η_c, η_m) and lower diagonal plots for NSGA-II with fixed parameter settings

The value of the adaptable parameter identified by the ANSGA-II for this problem is: $N = 200$. Figure 76 and diversity metric values for WATER in Table 29 show that the ANSGA-II with adaptable N finds a less spread of non-dominated solutions (less distribution of solutions on the Pareto-optimal fronts in upper diagonal plots of Figure 76(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) in solving the problem WATER than the NSGA-II with fixed parameter settings. The plots for ANSGA-II have less formed patterns than the plots for NSGA-II, implying that the ANSGA-II obtains less convergence than NSGA-II.

Results of ANSGA-II with Adaptable p_c, η_c, and Fixed N, p_m, η_m

This variant of ANSGA-II supports self-adaptive crossover probability (p_c) and self-adaptive crossover distribution index (η_c). The population size (N), mutation probability (p_m), and mutation distribution index (η_m) are set to the same parameter values used in the NSGA-II: $N = 100$, $p_m = 0.5$, and $\eta_m = 100$. In addition, the number of generations is set to the same values used in the NSGA-II: $G = 250$ for SCH, FON, POL, KUR, ZDT1, ZDT2, ZDT3, ZDT4, ZDT6; and $G = 500$ for DEB, SRN TNK, WATER. Table 30 presents performance results of the ANSGA-II with adaptable p_c, η_c against the original NSGA-II with fixed parameter settings on thirteen benchmark problems.

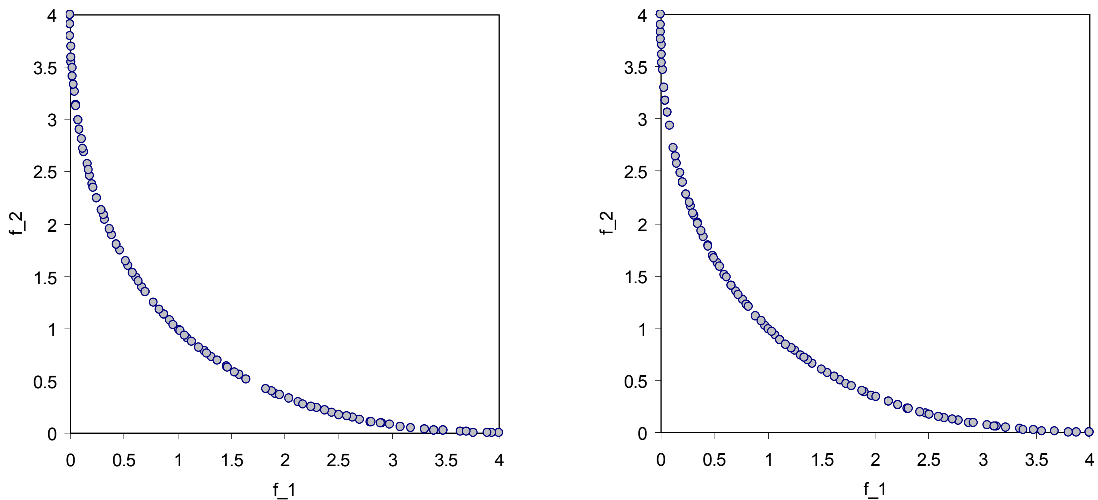
Table 30: Performance results of ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) against the original NSGA-II with fixed parameter settings

Problems	ANSGA-II						NSGA-II					
	N	G	Diversity Metric	Func. Eval.	Time (sec)	$N*G$ (1000)	N	G	Diversity Metric	Func. Eval.	Time (sec)	$N*G$ (1000)
SCH	100	250	0.5418	251	5	25	100	250	0.5711	251	4	25
FON	100	250	0.7097	251	5	25	100	250	0.7219	251	4	25
POL	100	250	0.9583	251	5	25	100	250	0.9538	251	3	25
KUR	100	250	0.8254	251	6	25	100	250	0.8121	251	4	25
ZDT1	100	250	0.7183	251	8	25	100	250	0.7431	251	4	25
ZDT2	100	250	0.7448	251	5	25	100	250	0.7390	251	6	25
ZDT3	100	250	0.8723	251	9	25	100	250	0.8731	251	5	25
ZDT4	100	250	0.7558	251	5	25	100	250	0.6766	251	5	25
ZDT6	100	250	0.7204	251	5	25	100	250	0.7046	251	8	25
DEB	100	500	0.8318	501	8	50	100	500	0.7772	501	15	50
SRN	100	500	0.7889	501	9	50	100	500	0.8011	501	16	50
TNK	100	500	0.8019	501	7	50	100	500	0.8072	501	8	50
WATER	100	500	0.7551	501	10	50	100	500	0.6274	501	10	50

For problems SCH, FON, ZDT1, ZDT4, SRN, and TNK, the ANSGA-II with adaptable p_c, η_c out-performs the original NSGA-II with fixed parameter settings in terms of finding better spread of non-dominated solutions and approximating to the true Pareto-optimal front. The ANSGA-II with adaptable p_c, η_c is able to find the global Pareto-optimal solutions for the problem ZDT4 while the NSGA-II with fixed parameter settings converges to a local Pareto-optimal front. This implies that adaptable crossover probability helps to improve performance on the problems that have many local Pareto-optimal fronts. For problems POL, ZDT2, ZDT3, and DEB, this variant of ANSGA-II performs very close to the NSGA-II with fixed parameter settings. Both this variant of ANSGA-II and the NSGA-II fail to converge to the global Pareto-optimal front on the problem ZDT6. For problems KUR and WATER, the NSGA-II with fixed parameter settings performs better. This variant has execution time comparable to that of required by the NSGA-II. This implies that the ANSGA-II is able to learn good values for the crossover parameters quickly. It also has execution time less than the variants with adaptable population size. This implies that the overhead for adapting population size is expensive. The plots of non-dominated solutions on thirteen benchmark problems obtained by the ANSGA-II and the NSGA-II are presented and compared in the following.

Results for the Two-Objective Test Problem SCH

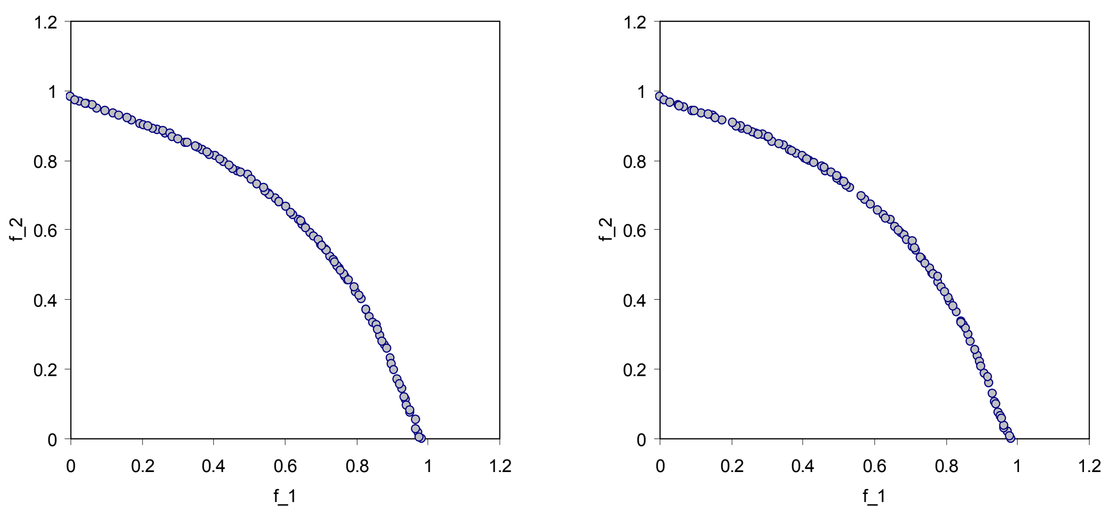
The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.8711 and average η_c = 89.72. Figure 77 and diversity metric values for SCH in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds a better distribution of Pareto-optimal solutions (distribution of solutions is a little bit more uniform in Figure 77(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem SCH than the NSGA-II with fixed parameter settings.



(a) (b)

Figure 77: Non-dominated solutions on SCH with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem FON



(a) (b)

Figure 78: Non-dominated solutions on FON with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.9685 and average η_c = 463.93. Figure 78 and diversity metric values for FON in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds a better distribution of Pareto-optimal solutions (distribution of solutions is a little bit more uniform in Figure 78(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem FON than the NSGA-II with fixed parameter settings.

Results for the Two-Objective Test Problem POL

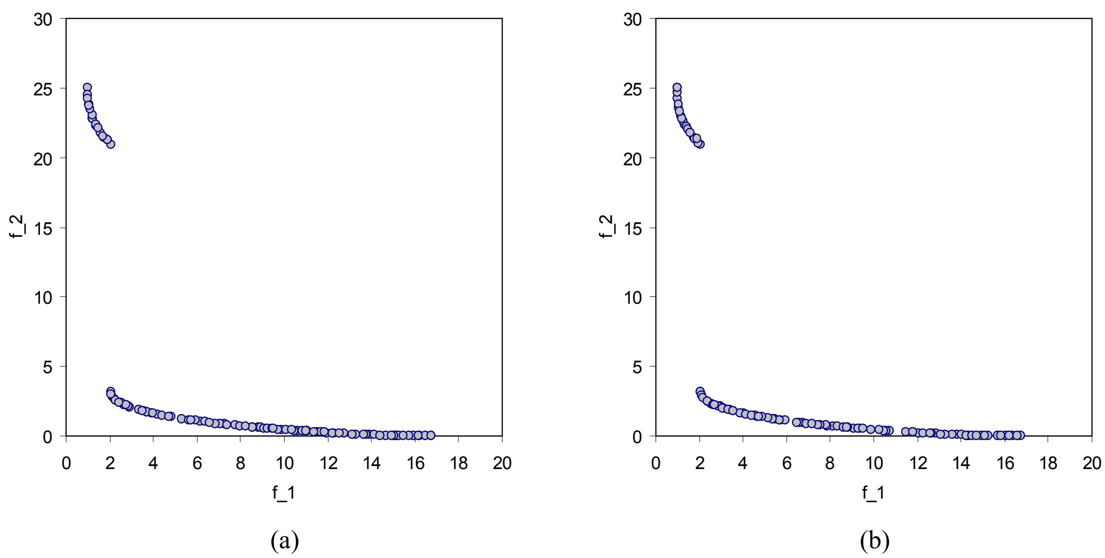


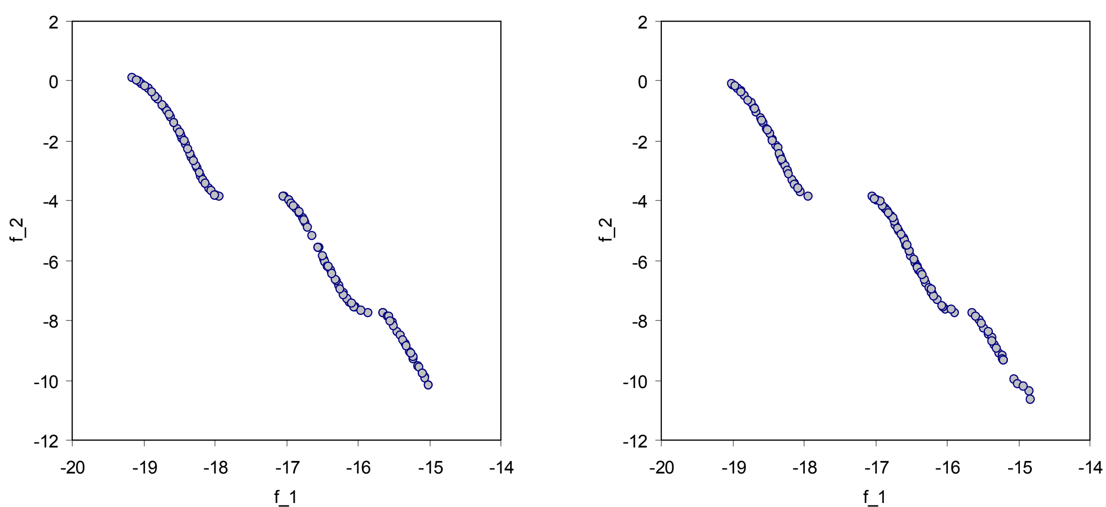
Figure 79: Non-dominated solutions on POL with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.6286 and average η_c = 334.08. Figure 79 and diversity metric values for POL in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds a compatible spread of Pareto-optimal solutions (distribution of solutions is about the same in Figure 79(a) and Figure 79(b), and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$ and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}}$ +

0.005) in solving the problem POL compared to the NSGA-II with fixed parameter settings.

Results for the Two-Objective Test Problem KUR

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.5072 and average η_c = 422.2. Figure 80(a) shows that the ANSGA-II with adaptable p_c, η_c performs worse than the NSGA-II because it is little bit short to obtain non-dominated solutions that cover the entire shape of the Pareto-optimal front (bottom right region). This implies that given the fixed population size and maximum number of generation (N = 100, G = 250), the ANSGA-II does not have enough time to learn good parameter values for p_c, η_c for the problem KUR.

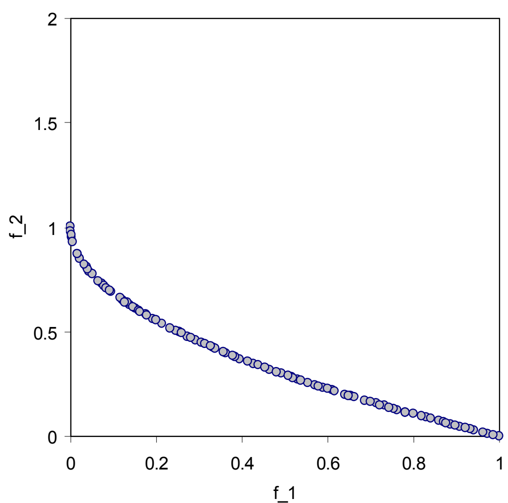


(a) (b)

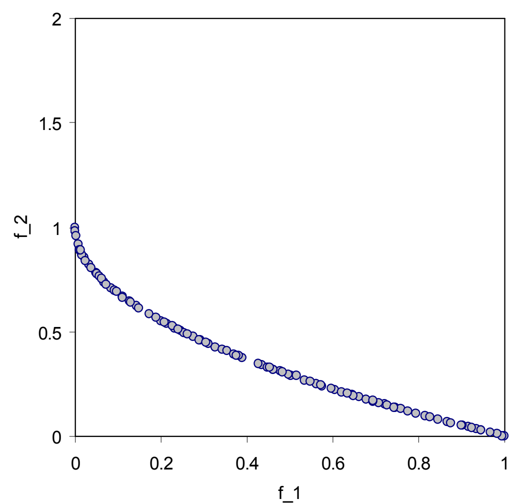
Figure 80: Non-dominated solutions on KUR with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT1

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.1982 and average η_c = 40.27. Figure 81 and diversity metric values for ZDT1 in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds a better distribution of Pareto-optimal solutions (less small gaps on the Pareto-optimal front of Figure 81(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem ZDT1 than the NSGA-II with fixed parameter settings.



(a)

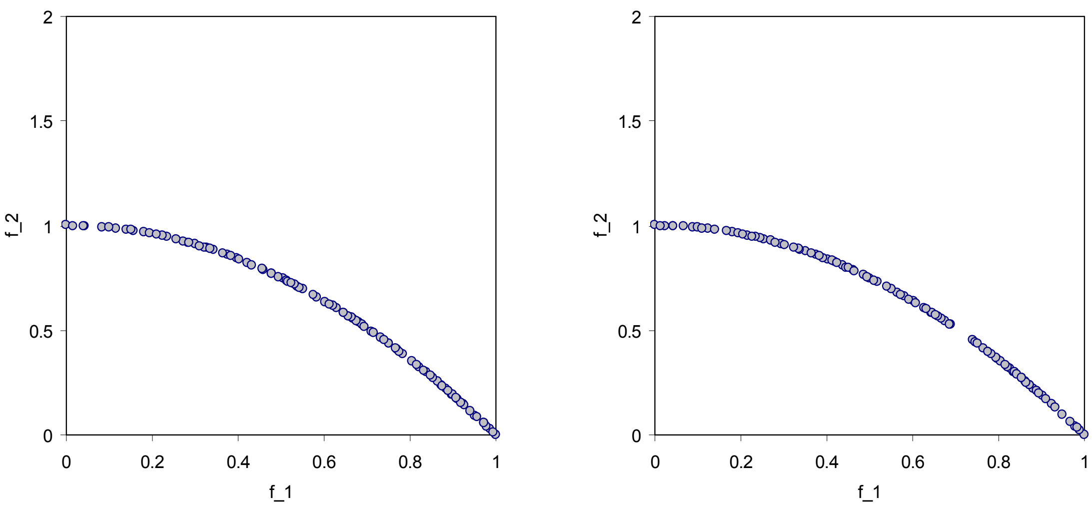


(b)

Figure 81: Non-dominated solutions on ZDT1 with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT2

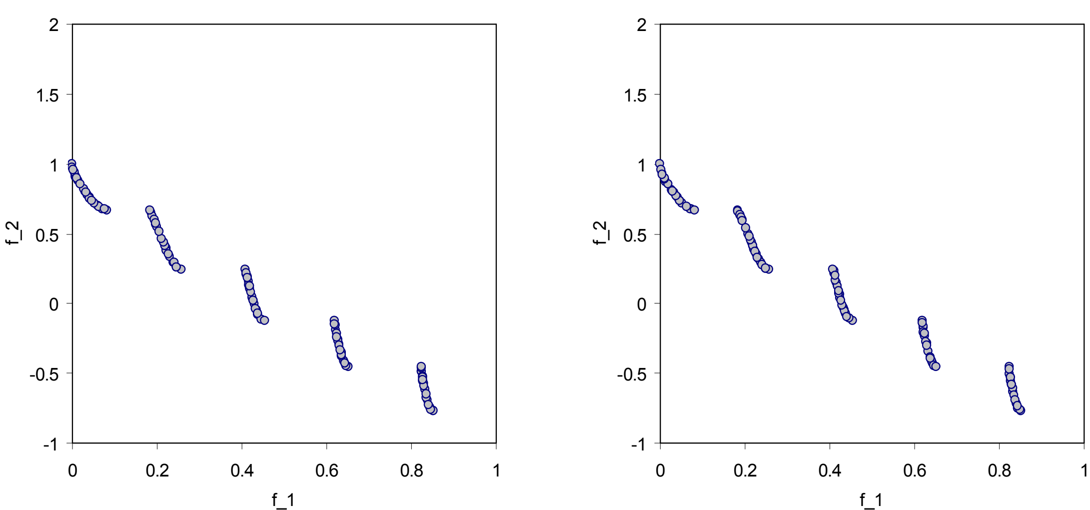
The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.1477 and average η_c = 200.27. Figure 82 and diversity metric values for ZDT2 in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds a less spread of Pareto-optimal solutions (more small gaps on the Pareto-optimal front in Figure 82(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) in solving the problem ZDT2. However, the distribution of Pareto-optimal solutions with ANSGA-II is adequate ($dm_{\text{ANSGA-II}} \leq 0.80$). This variant also requires a little bit less time than the NSGA-II to solve the problem ZDT2.



(a) (b)

Figure 82: Non-dominated solutions on ZDT2 with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT3



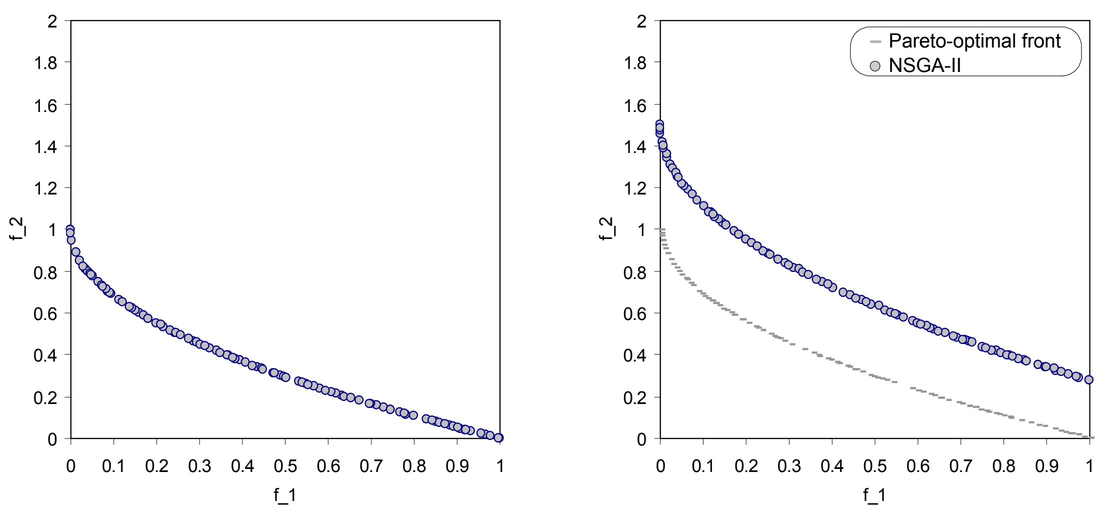
(a) (b)

Figure 83: Non-dominated solutions on ZDT3 with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.2987 and average η_c = 102.35. Figure 83 and diversity metric values for ZDT3 in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds the same spread of Pareto-optimal solutions (distribution of solutions is about the same in Figure 83(a) and Figure 83(b), and $dm_{\text{ANSGA-II}} < dm_{\text{NSGA-II}}$ and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}} - 0.001$) in solving the problem ZDT3 than the NSGA-II with fixed parameter settings.

Results for the Two-Objective Test Problem ZDT4

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.3072 and average η_c = 36.62. Figure 84 shows that the ANSGA-II with adaptable p_c, η_c is able to find the global Pareto-optimal solutions for the problem ZDT4 while the NSGA-II with fixed parameter settings fails to converge to the global Pareto-optimal front.

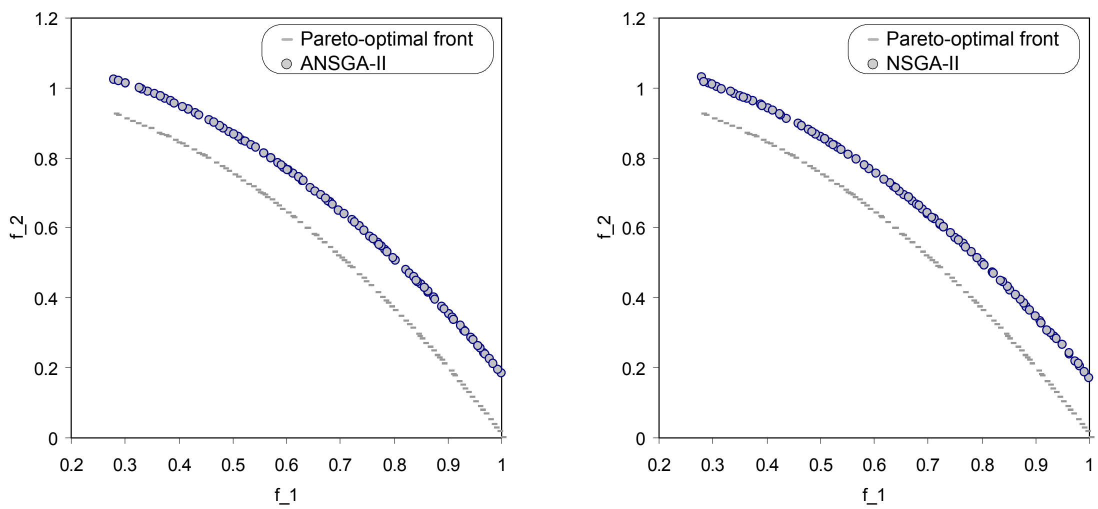


(a) (b)

Figure 84: Non-dominated solutions on ZDT4 with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT6

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.2381 and average η_c = 81.07. Figure 85 shows that both algorithms fail to converge to the global Pareto-optimal front.

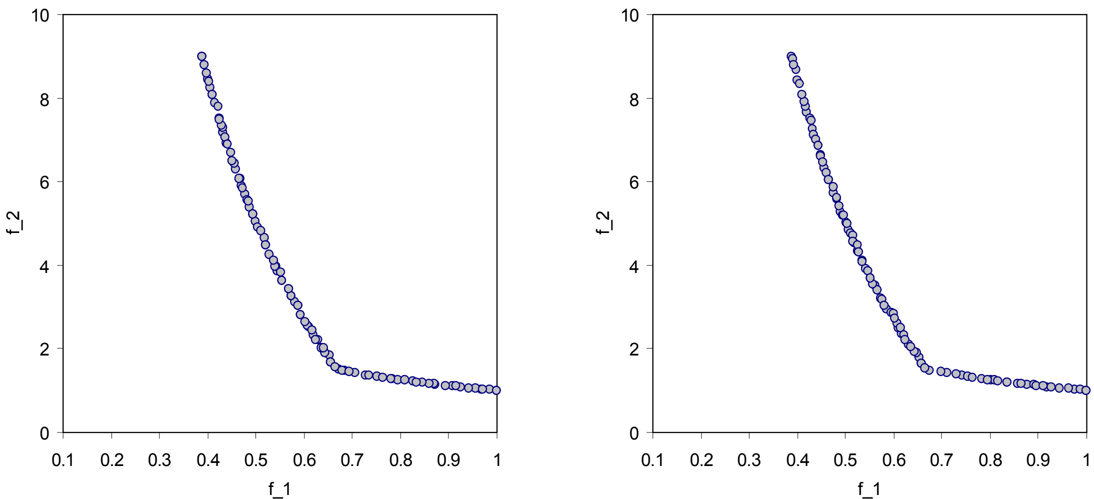


(a) (b)

Figure 85: Non-dominated solutions on ZDT6 with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem DEB

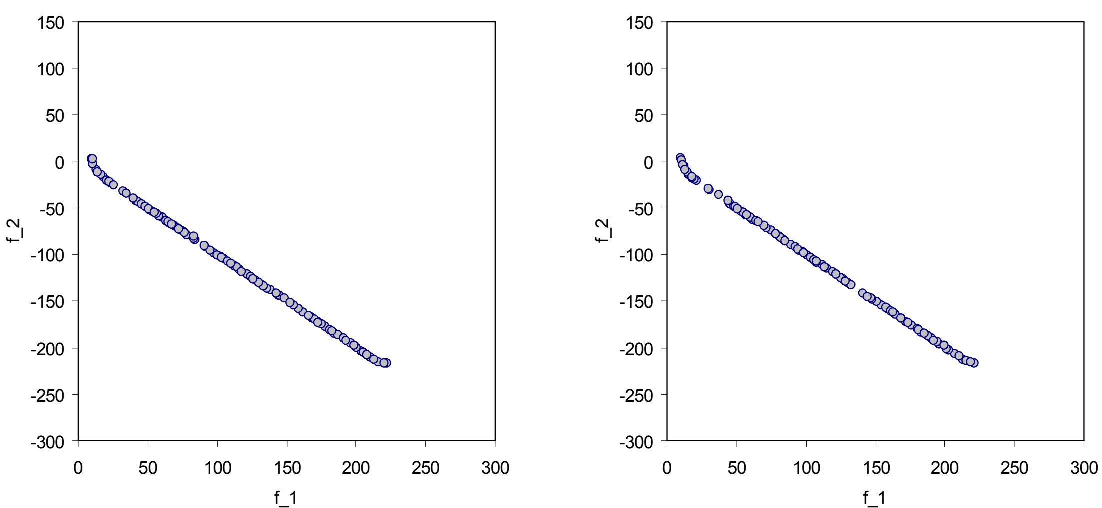
The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_c = 0.5814 and average η_c = 463.55. Diversity metric values for DEB in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds a less spread of Pareto-optimal solutions ($dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) in solving the problem DEB compared to that of the NSGA-II with fixed parameter settings. However, Figure 86 shows that two plots are very compatible. This is the only case in the results in which the diversity metric values appear to contradict with the plots. This implies that the diversity metric calculation is not very reliable. The ANSGA-II also takes less time to solve this problem compared to the NSGA-II. Therefore, it can be said that the non-dominated solution set obtained with this variant of the ANSGA-II is adequate.



(a) (b)

Figure 86: Non-dominated solutions on DEB with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem SRN

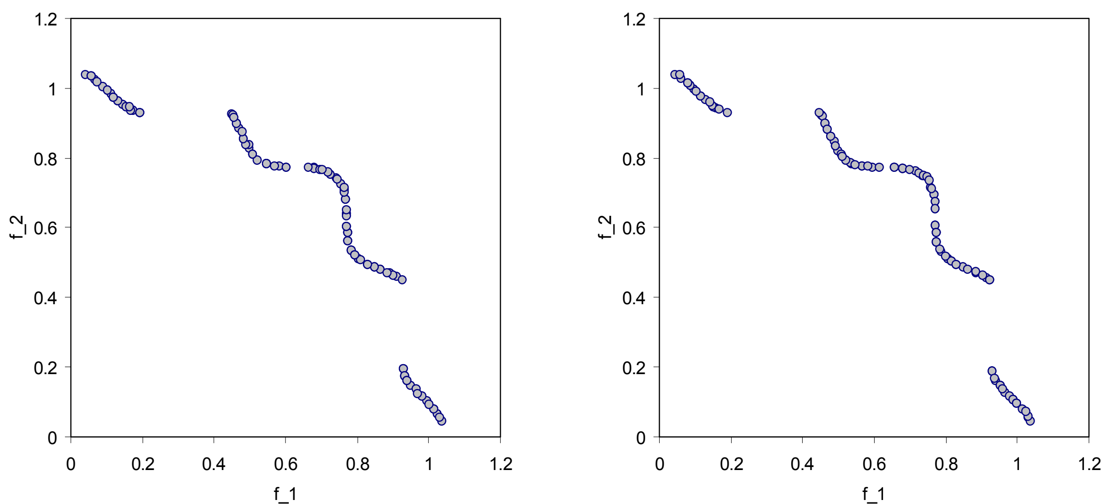


(a) (b)

Figure 87: Non-dominated solutions on SRN with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average $p_c = 0.5590$ and average $\eta_c = 451.28$. Figure 87 and diversity metric values for SRN in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds a better spread of Pareto-optimal solutions (less gaps on the Pareto-optimal front of Figure 87(a) $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$). The ANSGA-II also takes less time to solve this problem compared to the NSGA-II.

Results for the Two-Objective Test Problem TNK



(a) (b)

Figure 88: Non-dominated solutions on TNK with (a) ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average $p_c = 0.6958$ and average $\eta_c = 692.62$. Figure 88 and diversity metric values for TNK in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds a better spread of Pareto-optimal solutions (distribution of solutions is more continuous on the middle Pareto-optimal front of Figure 88(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem TNK than the NSGA-II with fixed parameter settings. The ANSGA-II also takes a little bit less time to solve this problem compared to the NSGA-II.

Results for the Five-Objective Real-World Problem WATER

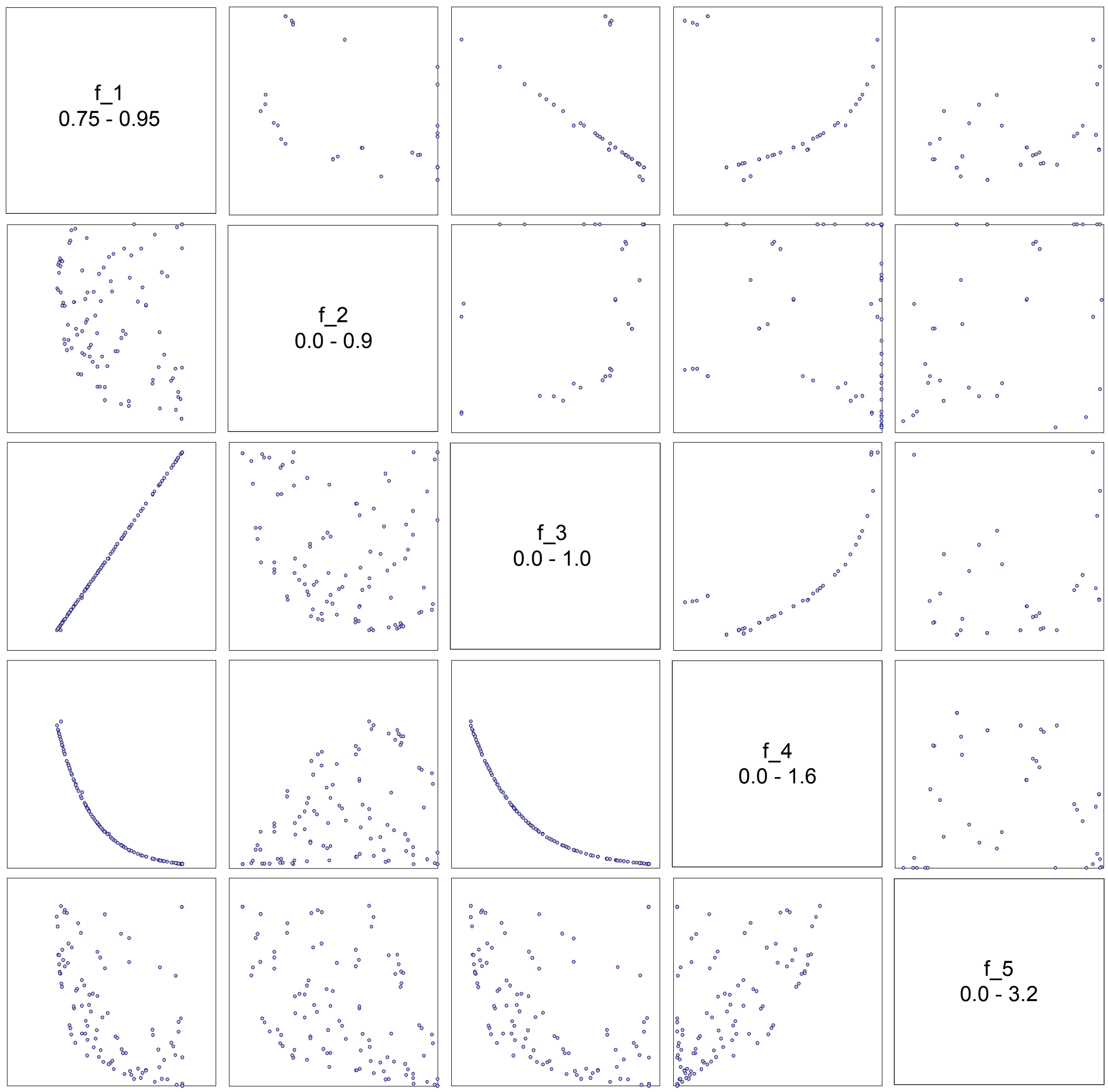


Figure 89: Non-dominated solutions on WATER with upper diagonal plots for ANSGA-II (adaptable p_c, η_c, and fixed N, p_m, η_m) and lower diagonal plots for NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average $p_c = 0.2940$ and average $\eta_c = 497.63$. Figure 89 and diversity metric values for WATER in Table 30 show that the ANSGA-II with adaptable p_c, η_c finds less spread of non-dominated solutions (less distribution of solutions on the Pareto-optimal fronts in upper diagonal plots of Figure 89(a) and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) in solving the problem WATER than the NSGA-II with fixed parameter settings. The plots for ANSGA-II have less formed patterns than the plots for NSGA-II, implying that this variant obtains less convergence than NSGA-II.

Results of ANSGA-II with Adaptable p_m, η_m, and Fixed N, p_c, η_c

This variant of ANSGA-II supports self-adaptive mutation probability (p_m) and self-adaptive mutation distribution index (η_m). The population size (N), crossover probability (p_c), and self-adaptive crossover distribution index (η_c) are set to the same parameter values used in the NSGA-II: N = 100, p_c = 0.9, and η_c = 20. In addition, the number of generations is set to the same values used in the NSGA-II: G = 250 for SCH, FON, POL, KUR, ZDT1, ZDT2, ZDT3, ZDT4, ZDT6; and G = 500 for DEB, SRN TNK, WATER. Table 31 presents performance results of the ANSGA-II with adaptable p_m, η_m against the original NSGA-II with fixed parameter settings on thirteen benchmark problems.

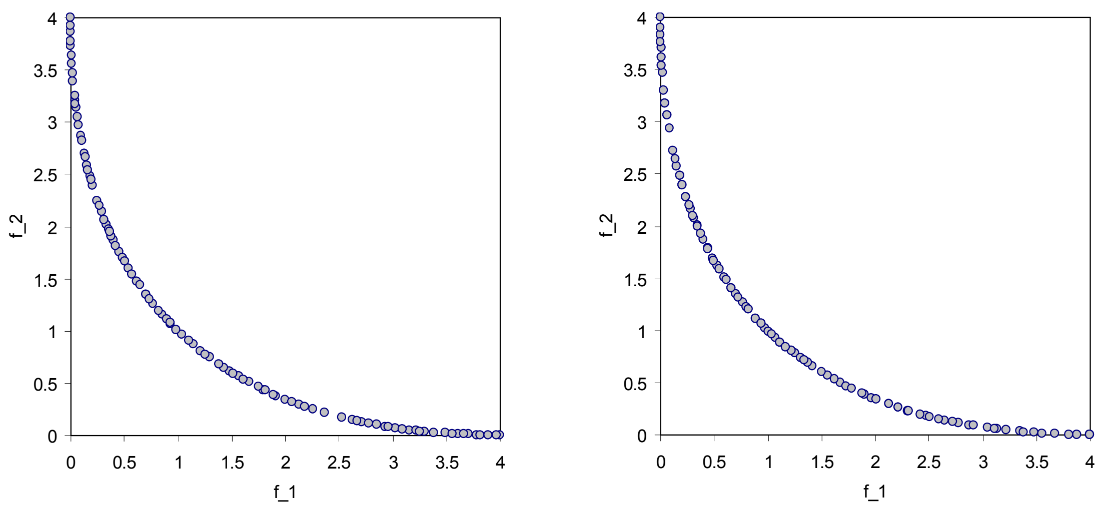
Table 31: Performance results of ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) against the original NSGA-II with fixed parameter settings

Problems	ANSGA-II						NSGA-II					
	N	G	Diversity Metric	Func. Eval.	Time (sec)	N*G (1000)	N	G	Diversity Metric	Func. Eval.	Time (sec)	N*G (1000)
SCH	100	250	0.5258	251	4	25	100	250	0.5711	251	4	25
FON	100	250	0.7051	251	5	25	100	250	0.7219	251	4	25
POL	100	250	0.9503	251	6	25	100	250	0.9538	251	3	25
KUR	100	250	0.8553	251	5	25	100	250	0.8121	251	4	25
ZDT1	100	250	0.7351	251	6	25	100	250	0.7431	251	4	25
ZDT2	100	250	0.7414	251	6	25	100	250	0.7390	251	6	25
ZDT3	100	250	0.7892	251	9	25	100	250	0.8731	251	5	25
ZDT4	100	250	0.7774	251	7	25	100	250	0.6766	251	5	25

ZDT6	100	250	0.6996	251	5	25	100	250	0.7046	251	8	25
DEB	100	500	0.7289	501	8	50	100	500	0.7772	501	15	50
SRN	100	500	0.7887	501	9	50	100	500	0.8011	501	16	50
TNK	100	500	0.8213	501	9	50	100	500	0.8072	501	8	50
WATER	100	500	0.5619	501	10	50	100	500	0.6274	501	10	50

For problems SCH, FON, ZDT1, ZDT3, ZDT4, DEB, and SRN, the ANSGA-II with adaptable p_m, η_m out-performs the original NSGA-II with fixed parameter settings in terms of finding better spread of non-dominated solutions and approximating to the true Pareto-optimal front. The ANSGA-II with adaptable p_m, η_m is able to find the global Pareto-optimal solutions for the problem ZDT4 while the NSGA-II with fixed parameter settings converges to a local Pareto-optimal front. This implies that adaptable mutation probability alone helps to improve performance on the problems that have many local Pareto-optimal fronts. For problems POL, ZDT2 and TNK, the ANSGA-II with adaptable p_m, η_m performs very close to the NSGA-II with fixed parameter settings. For problems KUR, ZDT6, and WATER, this variant of the ANSGA-II performs worse than the NSGA-II with fixed parameter settings. The most likely reason is that time spent on finding good parameter values for p_m, η_m is time taken away from finding diverse sets of non-dominated solutions. This variant has execution time comparable to that of required by the NSGA-II. This implies that the ANSGA-II is able to learn good values for the mutation parameters quickly. This variant has execution time less than the variants with adaptable population size. This implies that the overhead for adapting population size is expensive. The plots of non-dominated solutions on thirteen benchmark problems obtained by the ANSGA-II and the NSGA-II are presented and compared in the following.

Results for the Two-Objective Test Problem SCH



(a) (b)

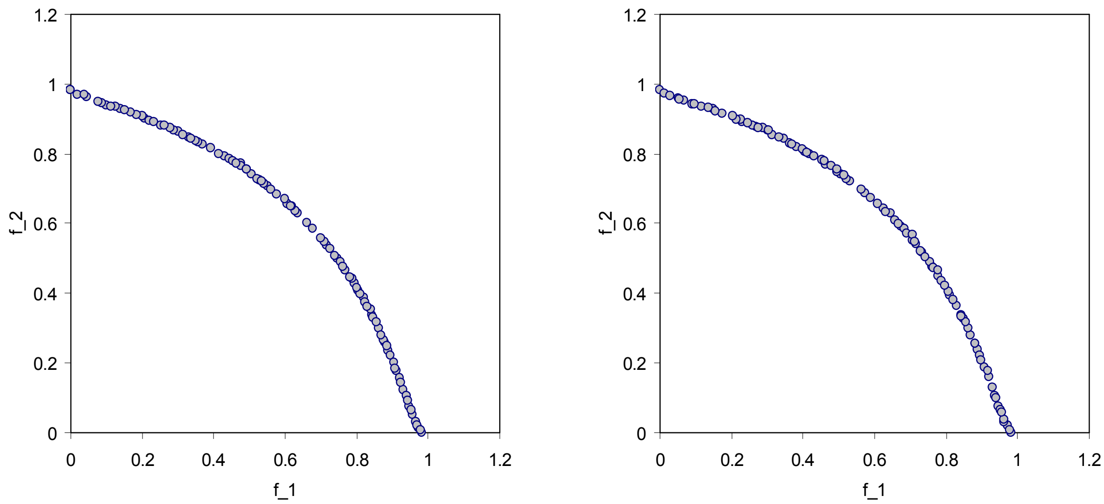
Figure 90: Non-dominated solutions on SCH with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.9173 and average η_m = 1541.05. Figure 90 and diversity metric values for SCH in Table 31 show that the ANSGA-II with adaptable p_m, η_m finds a better distribution of Pareto-optimal solutions (gaps are smaller in Figure 90(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem SCH than the NSGA-II with fixed parameter settings.

Results for the Two-Objective Test Problem FON

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.9067 and average η_m = 572.77. Figure 91 and diversity metric values for FON in Table 31 show that the ANSGA-II with adaptable p_m, η_m finds a better distribution of Pareto-optimal solutions (distribution of solutions is more continuous on the bottom area of the Pareto-optimal front of Figure 91(a) and $dm_{\text{ANSGA-II}}$

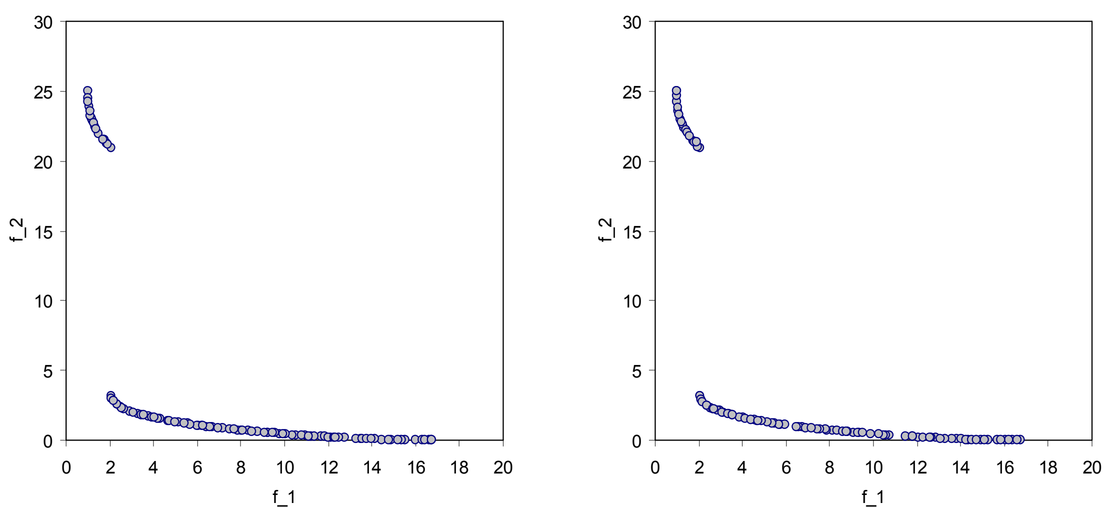
$\leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem FON than the NSGA-II with fixed parameter settings.



(a) (b)

Figure 91: Non-dominated solutions on FON with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem POL

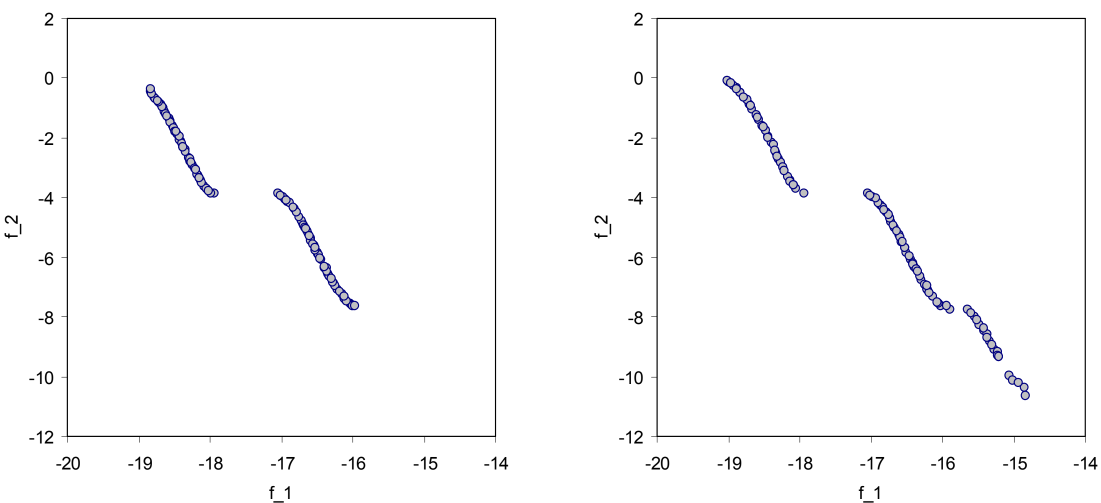


(a) (b)

Figure 92: Non-dominated solutions on POL with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.8779 and average η_m = 404.22. Figure 92 and diversity metric values for POL in Table 31 show that the ANSGA-II with adaptable p_m, η_m finds a better distribution of Pareto-optimal solutions (distribution of solutions is more continuous on the bottom Pareto-optimal front of Figure 92(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem FON than the NSGA-II with fixed parameter settings.

Results for the Two-Objective Test Problem KUR



(a) (b)

Figure 93: Non-dominated solutions on KUR with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.4169 and average η_m = 684.25. Figure 93 shows that the ANSGA-II with adaptable p_m, η_m performs worse than the NSGA-II because it fails to obtain non-dominated solutions that cover the entire shape of the Pareto-optimal front (bottom right region). This implies that given the fixed population size and number of generations (N = 100, G = 250), the ANSGA-II does not have enough time to learn good parameter values for p_m, η_m for the problem KUR.

Results for the Two-Objective Test Problem ZDT1

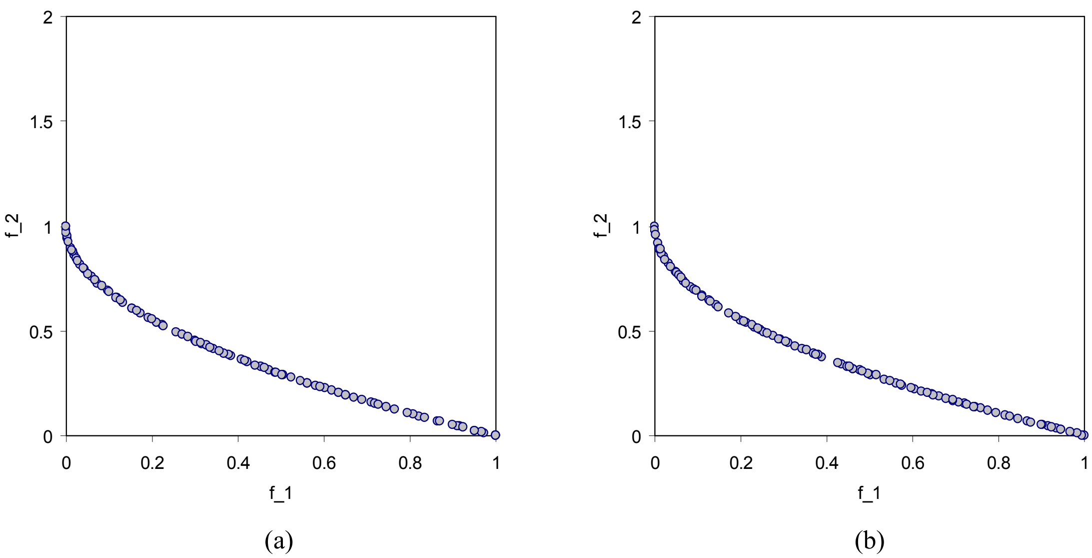
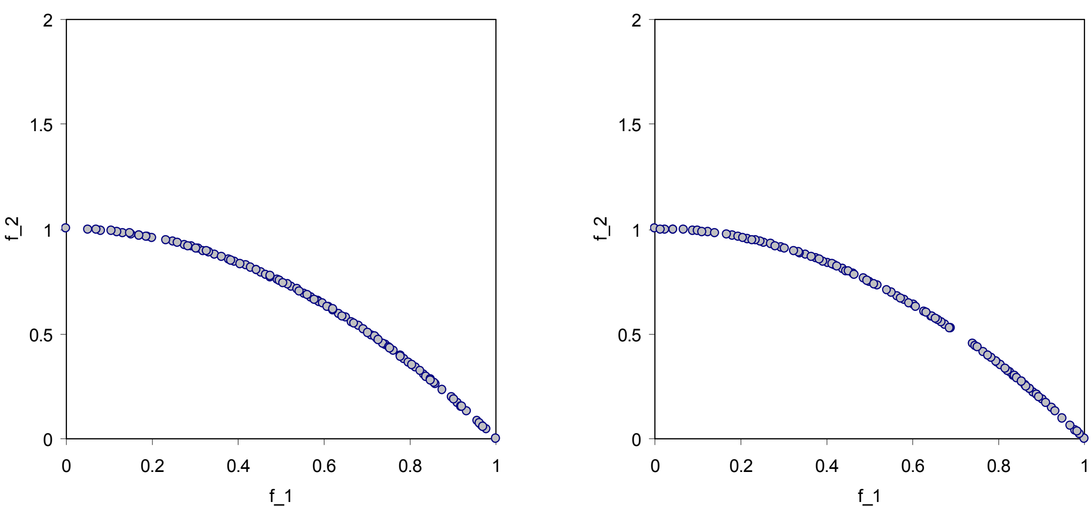


Figure 94: Non-dominated solutions on ZDT1 with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.0374 and average η_m = 43.91. Figure 94 and diversity metric values for ZDT1 in Table 31 show that the ANSGA-II with adaptable p_m, η_m finds a better distribution of Pareto-optimal solutions (gaps are smaller on the Pareto-optimal front of Figure 94(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem ZDT1 than the NSGA-II with fixed parameter settings.

Results for the Two-Objective Test Problem ZDT2

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.8030 and average η_m = 41.48. Figure 95 and diversity metric values for ZDT2 in Table 31 show that the ANSGA-II with adaptable p_m, η_m finds a compatible spread of Pareto-optimal solutions (the Pareto-optimal front of Figure 95(a) has a big gap at the top and the Pareto-optimal front of Figure 95(b) also has a big gap near the middle; $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$ and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} + 0.005$) in solving the problem ZDT2 compared to the NSGA-II with fixed parameter settings.

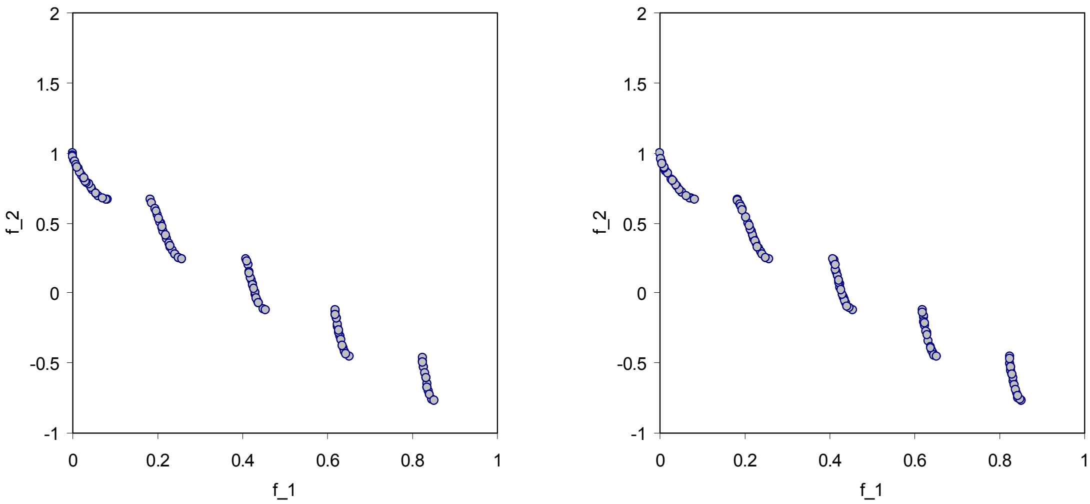


(a) (b)

Figure 95: Non-dominated solutions on ZDT2 with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT3

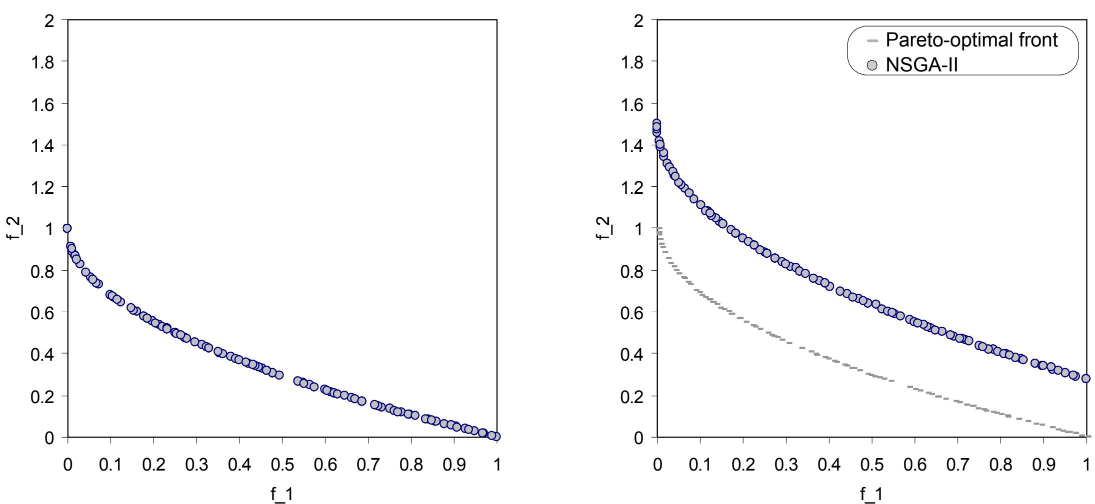
The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.0451 and average η_m = 165.89. Figure 96 and diversity metric values for ZDT3 in Table 31 show that the ANSGA-II with adaptable p_m, η_m finds a better distribution of Pareto-optimal solutions (distribution of solutions is about the same in Figure 96(a) and Figure 96(b) but $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem ZDT3 than the NSGA-II with fixed parameter settings.



(a) (b)

Figure 96: Non-dominated solutions on ZDT3 with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem ZDT4

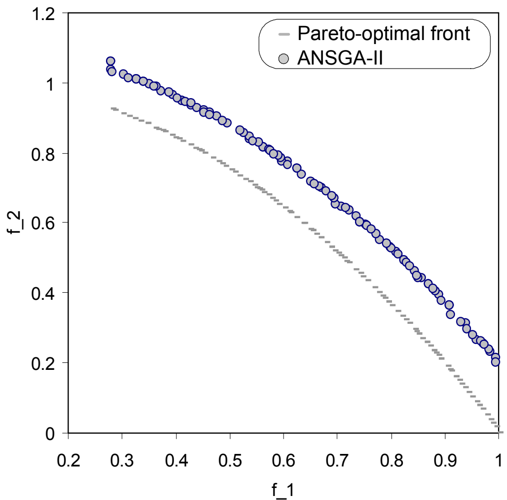


(a) (b)

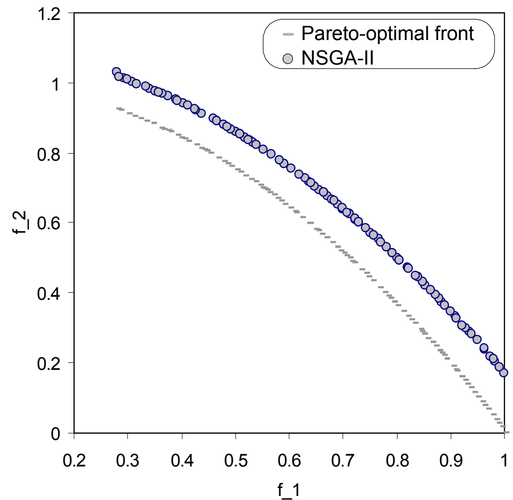
Figure 97: Non-dominated solutions on ZDT4 with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.4130 and average η_m = 168.37. Figure 97 shows that the ANSGA-II with adaptable p_m, η_m is able to converge to the global Pareto-optimal front for the problem ZDT4 while the NSGA-II with fixed parameter settings converges to a local Pareto-optimal front.

Results for the Two-Objective Test Problem ZDT6



(a)



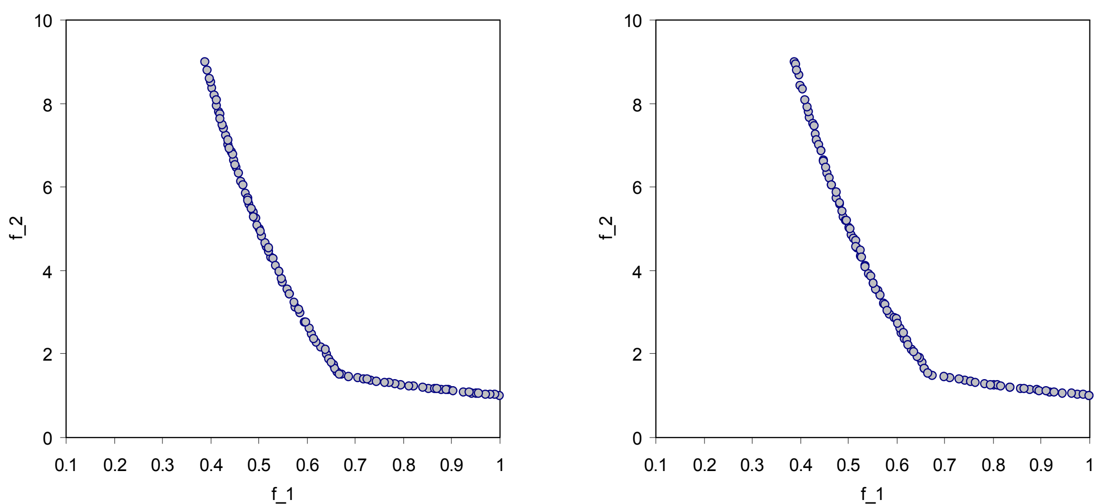
(b)

Figure 98: Non-dominated solutions on ZDT6 with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.0357 and average η_m = 54.03. Similar to the ANSGA-II with adaptable parameters N, p_c, p_m, η_c, η_m, Figure 98 shows that both the ANSGA-II with adaptable p_m, η_m and the NSGA-II with fixed parameter settings fail to converge to the global Pareto-optimal front for the problem ZDT6. Figure 98(a) also shows that this variant of ANSGA-II generates number non-dominated solutions that do not line up evenly on the local Pareto-optimal front.

Results for the Two-Objective Test Problem DEB

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.8346 and average η_m = 619.79. Figure 99 and diversity metric values for DEB in Table 31 show that the ANSGA-II with adaptable p_m, η_m finds a better distribution of Pareto-optimal solutions (distribution of solutions is more continuous on the Pareto-optimal front of Figure 99(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem DEB than the NSGA-II with fixed parameter settings. This variant also requires less time to solve the problem DEB compared to the NSGA-II.



(a) (b)

Figure 99: Non-dominated solutions on DEB with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem SRN

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.8150 and average η_m = 407.77. Figure 100 and diversity metric values for SRN in Table 31 show that the ANSGA-II with adaptable p_m, η_m finds a better distribution of Pareto-optimal solutions (less small gaps on the Pareto-optimal front of Figure 100(a) and $dm_{\text{ANSGA-II}} \leq dm_{\text{NSGA-II}} - 0.001$) in solving the problem SRN than the NSGA-II with fixed parameter settings. This variant also requires less time to solve the problem SRN compared to the NSGA-II.

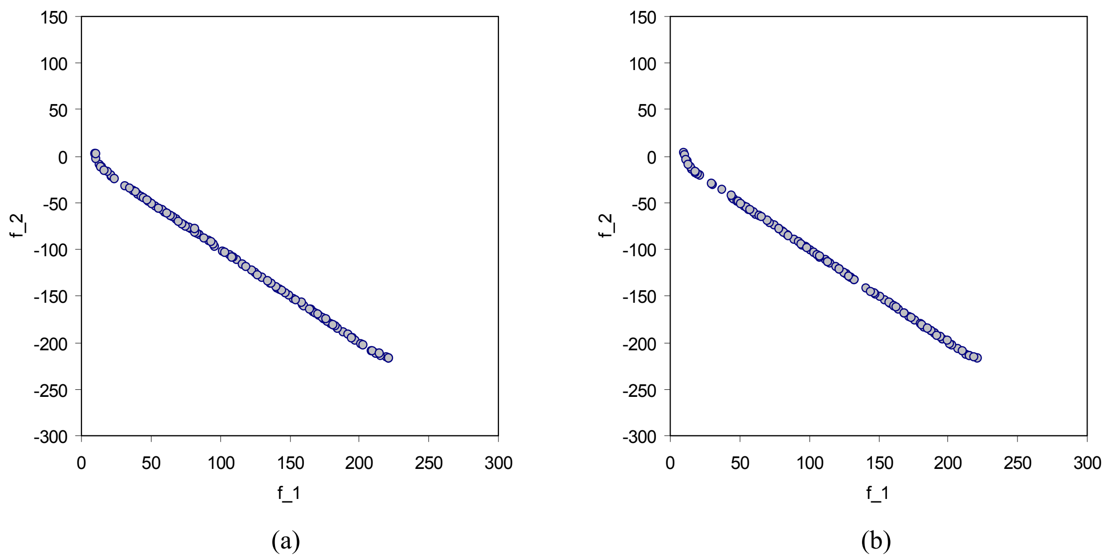


Figure 100: Non-dominated solutions on SRN with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

Results for the Two-Objective Test Problem TNK

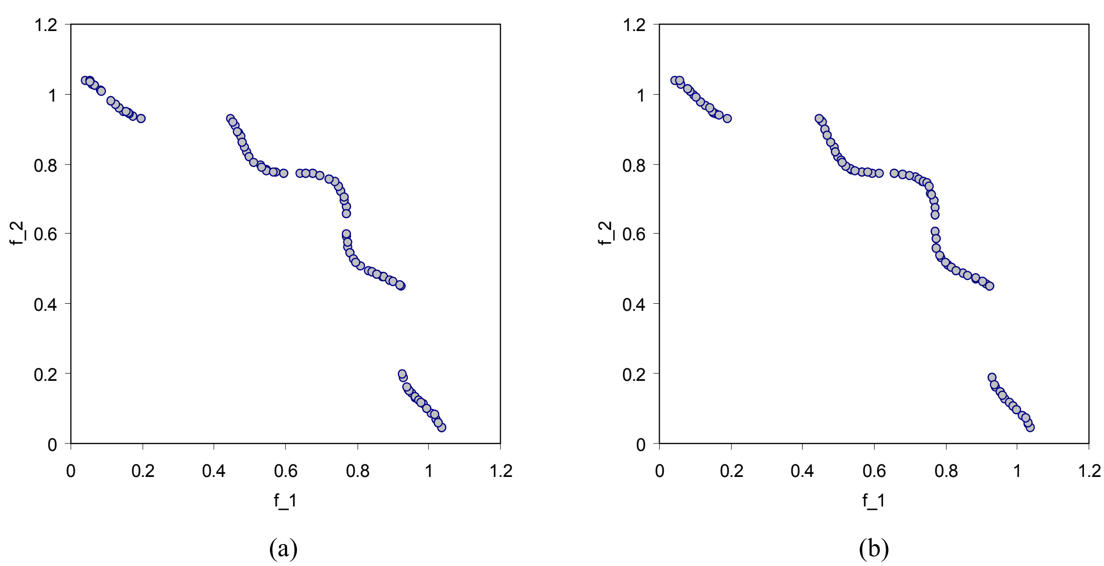


Figure 101: Non-dominated solutions on TNK with (a) ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and (b) NSGA-II with fixed parameter settings

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.5414 and average η_m = 893.99. Figure 101 and diversity metric values for TNK in Table 31 show that the ANSGA-II with adaptable p_m, η_m finds a less spread of Pareto-optimal solutions (more small gaps at the top and bottom of the Pareto-optimal fronts, and $dm_{\text{ANSGA-II}} > dm_{\text{NSGA-II}}$) in solving the problem TNK than the NSGA-II with fixed parameter settings. Since the scale on two axes is very small, it can be said that the distribution of non-dominated solution set with this variant of the ANSGA-II is adequate.

Results for the Five-Objective Real-World Problem WATER

The values of the adaptable parameters identified by the ANSGA-II for this problem are: average p_m = 0.8953 and average η_m = 322.92. Upper diagonal plots of Figure 102 show that the ANSGA-II with adaptable p_m, η_m finds less spread of non-dominated solutions in solving the problem WATER than the NSGA-II with fixed parameter settings. However, diversity metric values for WATER in Table 31 show $dm_{\text{ANSGA-II}} < dm_{\text{NSGA-II}}$, which is inaccurate. The reason is that the final metric value is calculated as the average of all metric values obtained on pairs of objective functions. This implies that the diversity metric calculation is not very reliable and visual observation is needed for the comparison of performance results. The plots for ANSGA-II have equivalent formed patterns than the plots for NSGA-II, implying that the ANSGA-II achieves the same convergence as the NSGA-II.

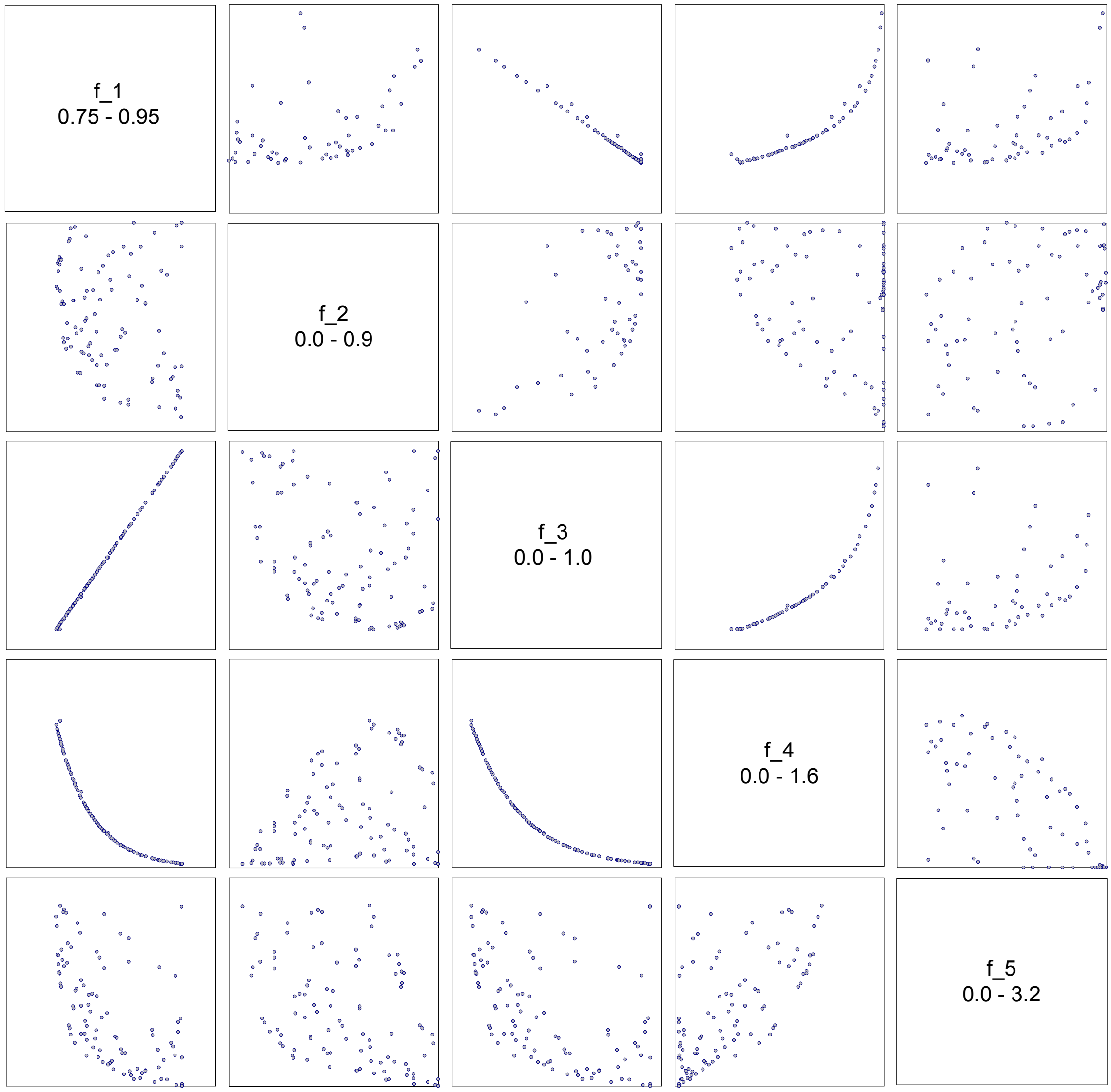


Figure 102: Non-dominated solutions on WATER with upper diagonal plots for ANSGA-II (adaptable p_m, η_m, and fixed N, p_c, η_c) and lower diagonal plots for NSGA-II with fixed parameter settings

Summary of Results

This chapter presented the results of the ANSGA-II with all three adaptable parameters (population size, crossover, mutation). In additions, the chapter presented the results of six variants of the ANSGA-II for studying the effect of using one or two adaptable parameters on the performance of the ANSGA-II. The performance of each variant of the ANSGA-II was evaluated and discussed by comparing its results obtained on thirteen benchmark multi-objective test problems with those obtained by the original NSGA-II. The results demonstrated that the ANSGA-II with adaptable N, p_c, and p_m is able to automate the process of selecting appropriate parameter values. It out-performs the original NSGA-II with fixed parameter settings and six other variants. The other six variants of the ANSGA-II perform worse than the original NSGA-II. The following ranks (1 for being the best) can be classified on the algorithms: (1) ANSGA-II with all three adaptable parameters N, p_c, p_m; (2) NSGA-II using original parameter settings, (3) ANSGA-II with adaptable mutation probability alone; (4) ANSGA-II with adaptable crossover probability alone; (5) ANSGA-II with adaptable crossover probability and mutation probability; (6) ANSGA-II with adaptable population size and crossover probability; (7) ANSGA-II with adaptable population size and mutation probability; (8) and ANSGA-II with adaptable population size only.

The conclusions on the results of this research, implications of the study, recommendations for further research, and summary of this research are presented in the next chapter.

Chapter 5

Conclusions, Implications, Recommendations, and Summary

Conclusions

The ANSGA-II was developed in an effort to apply parameter control techniques to a MOEA named NSGA-II. The performances of the ANSGA-II and its six variants were evaluated and discussed by comparing the results obtained by each variant on thirteen benchmark multi-objective test problems with those obtained by the original NSGA-II. This study answers the research questions, which were raised in Chapter 1, as described in the following:

The ANSGA-II with all three adaptable parameters (N, p_c, and p_m) is able to automate the process of selecting appropriate parameter values. It is able to find good values for these parameters quickly during its run. It out-performs the original NSGA-II with fixed parameter settings and six variants of the ANSGA-II in terms of finding a diverse set of non-dominated solutions and converging close to the true Pareto-optimal front. The ANSGA-II solves easy problems using smaller population sizes than those of required by the NSGA-II and it is able to find better solutions than the NSGA-II. On difficult problems, the ANSGA-II is able to find better solutions than the NSGA-II by increasing population sizes to accommodate the problem difficulty. On most problems, the improvement comes with the cost of longer execution time due to overheads of solving the problem and learning good parameter values at the same time. This means that the number of generations and the number of function evaluations are higher than those of required by the NSGA-II. However, the execution time appears to be acceptable on all thirteen benchmark multi-objective test problems.

There is no definite conclusion on which adaptable parameter among three adaptable parameters (N, p_c, p_m) affects the performance of the ANSGA-II the most or the least. In general, the variants of ANSGA-II with one or two adaptable parameters among three parameters N, p_c, and p_m out-perform the original NSGA-II on easy problems. On difficult problems, they have mixed performance results. Either adaptable p_c or p_m alone enables the algorithm to converge to the global Pareto-optimal front on the problem ZDT4 while the original NSGA-II is trapped in a local Pareto-optimal front. Other variants (adaptable N and p_c, adaptable N and p_m, adaptable p_c and p_m, and adaptable N alone) also fail to converge to the global Pareto-optimal front on the problem ZDT4. The reason for these later variants fail to converge to the true Pareto-optimal front is that there is not enough diversity in the population but the distribution of the obtained solutions is good; therefore, the algorithm terminates prematurely. On complex problems with several discontinuous Pareto-optimal fronts such as KUR, adaptable p_c alone or p_m alone fails to obtain non-dominated solutions that cover the entire shape of the Pareto-optimal fronts. The most likely reason is that the algorithm does not have enough time to learn good parameter values given insufficient population size and number of generations. However, the ANSGA-II with adaptable N, p_c and the ANSGA-II with adaptable N perform better than the original NSGA-II on the problem KUR.

Regarding the overhead for adapting parameters and solving the problem at the same time, the variants with adaptable population size N take longer time than other variants without adaptable N due to overheads of executing multiple populations simultaneously for learning a proper population size. Other variants without adaptable population size have execution time comparable to that of required by the NSGA-II. This implies that the ANSGA-II is able to learn good values for crossover and mutation parameters quickly and the cost for adapting population size is expensive.

The running convergence metric using the population-agglomeration technique suggested by Deb & Jain (2002) was integrated into the ANSGA-II and evaluated. This metric is inefficient and unreliable for the ANSGA-II. The metric works well when the Pareto-optimal solution set for the problem being solved is known in advance. However, the Pareto-optimal set is usually unknown in advance for real-world problems and the ANSGA-II is only useful if it can solve problems with unknown Pareto-optimal sets. The population-agglomeration technique can be used in combination with the running convergent metric to handle the case with unknown Pareto-optimal set. When used in the ANSGA-II, this technique requires extensive memory and computational resources. Multiple reference sets (with their size increasing with population sizes and number of generations) must be maintained for multiple populations. As the population size is getting bigger, the calculation of this technique is unacceptable slow in later generations of a population. Moreover, the technique relies merely on the fact that the algorithm being used is able to approximate to the true Pareto-optimal front eventually; otherwise, the reference set becomes useless. Therefore, the running convergent metric is not used in the ANSGA-II. A simple work-around technique is used instead. This technique simply allows the ANSGA-II to run for a while until the number of non-dominated solutions in the first rank at least equal to the required minimum number of solutions (i.e. the initial population size), and then the algorithm starts to calculate the diversity metric. This work-around technique appears to work effectively.

The ANSGA-II adopts the running diversity metric for measuring the diversity of the obtained solutions suggested by Deb et al. (2002). This diversity was modified to handle unknown Pareto-optimal set and problems with more than two objectives. The metric has a difficulty in distinguishing between valid gaps in problems with several discontinuous Pareto-optimal fronts and invalid gaps in problems with continuous Pareto-optimal fronts. However, the modified running diversity metric appears to work effectively in comparing two or more non-dominated solution sets among different populations during the execution of the ANSGA-II and it enables the ANSGA-II to select a proper population size for the problem being solved.

Until there exists an efficient and reliable running convergent metric, the parameter control techniques should be applied to MOEAs, which have been verified that they can find diverse non-dominated solution sets with good convergence on benchmark test problems borrowed from the MOEA literature. The parameter control techniques can be applied effectively to any MOEA that has a proof of convergence to the true Pareto-optimal set while preserving diversity of the obtained solutions at the same time (e.g. ε-MOEA).

Implications

MOEAs will continue to be used increasingly in a wide range of real-world multi-objective optimization applications due to their capability to find several good trade-off solutions for all objectives in a single run of the algorithm. Today, the MOEA repository (http://www.lania.mx/~ccoello/EMOO/) contains over 2178 papers, from which a vast majority are applications (Coello, 2005). A robust, efficient, and easy to use MOEA such as the ANSGA-II encourages more practitioners to use MOEA to solve real-world multi-objective optimization problems. The outcomes of this dissertation demonstrate that parameter control techniques indeed help to improve a MOEA's performance (NSGA-II specifically) in terms of finding a diverse set of non-dominated solutions and converging close to the true Pareto-optimal front. Therefore, more research should be done in this area.

Recommendations

The following additional studies and extensions are recommended and discussed briefly:

- Future research will be directed to applying and verifying the ANSGA-II on more real-world problems such as telecommunication network design (Flores et al., 2003; Maple et al., 2004), software quality enhancement (Khoshgoftar, 2004), risk-based corrective action design (Gopalakrishnan et al., 2001).
- Applying parameter control techniques used in ANSGA-II to other MOEAs such as ε-MOEA: The ε-MOEA was developed by Deb, Mohan, & Mishra (2003) based on the ε-dominance concept (see Definition of Terms). The ε-dominance concept requires the user to define the precision with which they want to evaluate each objective by specifying an appropriate ε value for each objective. Therefore, the algorithm introduces a new user-defined parameter: the ε-vector. However, the ε values can be implemented as self-adaptive

parameter as well (Laumanns et al., 2002). The ε-MOEA uses the ε values to find an approximation of Pareto-optimal set that meets the user-defined precisions. The ε-MOEA has been proven to find well-converged and well-distributed solutions with a much less computational effort when compared to other popular MOEAs such as NSGA-II, SPEA2, and PESA. Moreover, a ε-dominance based MOEA has a proof of convergence to the true Pareto-optimal set while preserving diversity of the obtained solutions at the same time (Laumanns et al., 2002). In addition, the archive size can be calculated based on the ε-vector (Laumanns et al., 2002).

- Develop better performance metrics: The parameter control techniques in MOEA rely on the performance metrics to monitor the progress of the MOEA during its run in order to adjust the values of its parameters accordingly. Therefore, more reliable and efficient performance metrics are needed, especially for problems with more than two objectives.

Summary

MOEAs are not easy to use because they require parameter tunings of three main parameters - population size, crossover probability, and mutation probability - in order to achieve the desirable solutions and performance for an arbitrary complex problem. The task of tuning these parameters is not trivial due to the complex and nonlinear interactions among the parameters and their dependency on many aspects of the particular problem being solved such as the search space size and the shape of the fitness surface. Moreover, the use of fixed parameter settings may lead to slow convergence and sub-optimal obtained solutions (i.e. solutions are not well-spread and not close to the true Pareto-optimal front), especially when large search spaces are to be explored in solving complex optimization problems because the proper parameter values are not fixed but varied during a run of a MOEA.

This dissertation aims to investigate simultaneous parameter control techniques in MOEA for all three parameters - population size, crossover, and mutation. The goal of this dissertation is to develop a MOEA with adaptable population size, crossover, and mutation, for automating the process of selecting appropriate parameter values in order to make the MOEA more efficient, easier to use and available to more users. This MOEA is built on the NSGA-II (Non-dominated Sorting Genetic Algorithm II), which supports static parameters, and named as ANSGA-II (Adaptable NSGA-II).

The dissertation uses evaluation research method, which consists of the following main steps. Formative studies of existing parameter control techniques and the NSGA-II are performed to identify the available techniques that can be used, issues and barriers that are needed to be resolved. The new algorithm ANSGA-II is then developed. The ANSGA-II is evaluated against the original NSGA-II using the same benchmark multi-objective problems that were used in the study of the original NSGA-II. Since the same benchmark problems are used, the results generated by the ANSGA-II can be easily compared to those of the NSGA-II for validation.

Several parameter control methods have been proposed and applied successfully for single objective optimization problems using simple GAs. One of the most significant empirical studies was performed by Bäck, Eiben, & van der Vaart (2000) in which simple

GAs have one or all three parameters (N, p_c, and p_m) adjusted during the run. The results of this study show the superiority of the GA with adaptable parameters.

Most MOEAs such as NSGA-II, PAES, and SPEA2 support fixed parameter settings. Some previous studies have applied parameter control techniques to MOEAs. However, these studies focus on one or two parameters in isolation and ignore other parameters. These studies have shown that parameter control techniques used in single-objective GA work differently in the multi-objective cases (Laumanns et al., 2001; Tran, 2005). In contrast to single-objective optimization, where objective function and fitness function are often the same, in multi-objective optimization, both fitness assignment and selection must support several objectives. The result of the multi-objective optimization process is usually not a single solutions but a set of trade-off solutions. These trade-off solutions converge towards different areas of the Pareto-optimal front and proper parameter values differ between these solutions. Moreover, in a MOEA, each solution is assigned a fitness value equal to its non-dominated rank in the population. This fitness assignment imposes a barrier in comparing two different non-dominated solutions set. In order to adjust the values of parameters, the progress of a MOEA run must be monitored and evaluated, which involves comparing non-dominated solution sets among generations to see how the obtained solutions vary with generations. However, all of these good solutions are in the first non-dominated front and have the same rank value. As a result, it is difficult to determine the better non-dominated solution set between two sets of non-dominated solutions. Performance metrics can be integrated into a MOEA to measure the convergence and diversity of the obtained solutions during its run in order to monitor and provide the MOEA's progress for adjusting the values of parameters. These running performance metrics should be reliable and efficient in order to provide correct progress information without spending too much time on metric calculations and taking away time for finding the solutions. Several performance metrics have been introduced in the MOEA literature. But most of these metrics are applicable to two-objective problems and inefficient for using as running metrics (Deb & Jain, 2002).

The NSGA-II, which the ANSGA-II is built upon, is one of the best-known MOEAs. The algorithm has been recognized to perform as well or better than other MOEAs with the same goal of finding a diverse Pareto-optimal solution set such as the PAES (Knowles & Corne, 1999) and SPEA2 (Zitzler et al., 2002). The major features of NSGA-II include low computational complexity, parameter-less diversity preservation, elitism, and real-valued representation. The NSGA-II uses a real-coded simulated binary crossover (SBX) operator and a real-coded polynomial mutation operator to support crossover and mutation operations directly to real-valued decision variables. The SBX operator introduces an additional user-defined parameter: the crossover distribution index η_c, which affects the probability distribution of the SBX operator. Likewise, the polynomial mutation operator introduces an additional user-defined parameter: the mutation distribution index η_m, which affects the probability distribution of the polynomial mutation operator. Deb (2001) pointed out that convergence cannot be guaranteed with NSGA-II because Pareto-optimal solutions may be replaced by other inferior non-dominated solutions due to the way the algorithm preserve elitism. Elitism in NSGA-II is ensured by comparing the current population with previously found best non-dominated solutions and by combining the parent and child populations to form a combined population with size 2*N*. The combined population is then sorted according to non-domination. As long as the size of the first non-dominated set is not larger than the population size, the algorithm preserves all of them in the new population of size N. However, in later generation, when the first non-dominated set has nearly converged to the Pareto-optimal set, there might be more than N solutions in the first non-dominated set of the combined parent-offspring population, and only those solutions with greater crowding distance (less crowded area) are chosen. In doing so, the algorithm has no way to know which solutions are already Pareto-optimal and which are not Pareto-optimal (but non-dominated). As a result, already found Pareto-optimal solutions may be replaced by other inferior non-dominated solutions and convergence cannot be guaranteed.

In the ideal approach to multi-objective optimization, there are two tasks: minimize the distance of the obtained solutions to the Pareto-optimal set and maximize the diversity of the obtained non-dominated set. Despite the fact that many new and improved MOEAs have been introduced, there severely lack for studies related to theoretical convergence analysis with guaranteed diversity of solutions in MOEAs (Deb, 2001; Laumanns et al., 2002). In this regard, several studies have proposed a number of MOEAs, which ensure convergence to the true Pareto-optimal set but do not guarantee the diversity of the obtained non-dominated set (Rudolph, 1998; Veldhuizen & Lamont, 1998; Hanne, 2000b, 2000a; Rudolph & Agapie, 2000; Rudolph, 2001). Since achievement of convergence does not automatically guarantee achievement of diversity. Therefore, it is also necessary to have a proof of diversity of the obtained non-dominated set. Until recently, Laumanns et al. (2002) proposed a new class of MOEAs based on the ε-dominance concept which have both properties of convergence to the true Pareto-optimal set and diversity of the obtained non-dominated set together. They also provided a proof of convergence to the true Pareto-optimal set while preserving diversity of the obtained solutions at the same time.

In the ANSGA-II, the crossover and mutation parameters are attached to each solution in the population and allowed to co-evolve with each solution. This enables the algorithm to carry prior successful crossover and mutation for creating children solutions and for adaptation of these two parameters since good crossover and mutation probabilities are associated with good candidate solutions. The ANSGA-II determines a proper population size by running several populations with different population sizes simultaneously (adopting the multiple population approach of Harik & Lobo (1999)). Two running metrics for measuring convergence and diversity of non-dominated solution sets are integrated into the ANSGA-II and investigated for their effective use in comparing non-dominated solution sets among different populations during the execution of the ANSGA-II. The idea is that when the ANSGA-II obtains a diverse non-dominated solution set with good convergence among different populations, it can terminate with a proper population size.

This dissertation investigates the convergent metric, which can work with an unknown set of Pareto-optimal solutions by using a population-agglomeration technique, suggested by Deb & Jain (2002). The conclusion is that the running convergent metric is not reliable and efficient for the ANSGA-II. In the population-agglomeration technique, a reference set is defined as the non-dominated set of all combined non-dominated sets from previous generations up to the current generation. The convergent metric for a population is the calculated by finding the smallest normalized Euclidean distance among all distances from each non-dominated solution in the population to each solution in the reference set. In the ANSGA-II, this technique requires extensive memory and computational resources. Multiple reference sets (with their sizes increasing with population sizes and number of generations) must be maintained for multiple populations. As the population size is getting bigger, the calculation of this technique is unacceptable slow in later generations of a population. Moreover, the technique relies merely on the fact that the algorithm being used is able to approximate to the true Pareto-optimal front eventually; otherwise, the reference set becomes useless. Therefore, the running convergent metric is not used in the ANSGA-II and a simple work-around technique is used instead. This technique simply allows the ANSGA-II to run for a while until the number of non-dominated solutions in the first rank at least equal to the required minimum number of solutions (initial population size), and then the algorithm starts to calculate the diversity metric. This work-around technique appears to work effectively. It should also be emphasized that the parameter control techniques should be applied to good MOEAs, which have been verified that they can find diverse non-dominated solution sets with good convergence on benchmark test problems borrowed from the MOEA literature.

In the ANSGA-II, the diversity metric, which was used in the original study of NSGA-II (Deb, Pratap et al., 2002), is modified to handle unknown Pareto-optimal set and problems with more than two objectives. The Euclidean distances between the known Pareto-optimal extreme solutions and the boundary solutions of the obtained non-dominated set are replaced with the Euclidean distances between the extreme boundary solutions (solutions with smallest and largest function values) and the boundary solutions of the obtained non-dominated set. The diversity metric value for problems with more than two objectives is calculated as the average diversity metric value of all diversity metric values calculated on combinations of objective function pairs. For example, the five-objective problem WATER has ten different pairs of objective functions and the diversity metric value Δ is calculated as: $\Delta = [\Delta (f_1, f_2) + \Delta (f_1, f_3) + \Delta (f_1, f_4) + \Delta (f_1, f_5) + \Delta (f_2, f_3) + \Delta (f_2, f_4) + \Delta (f_2, f_5) + \Delta (f_3, f_4) + \Delta (f_3, f_5) + \Delta (f_4, f_5)] / 10)$. The diversity metric has a difficulty in distinguishing between valid gaps in problems with discontinuous Pareto-optimal fronts and invalid gaps in problems with continuous Pareto-optimal fronts. However, the modified running diversity metric appears to work effectively in comparing two or more non-dominated solution sets during the execution of the ANSGA-II and it enables the ANSGA-II to select proper population sizes for the problems being solved.

The results demonstrates that the goal of this dissertation has been achieved: the ANSGA-II with adaptable parameters crossover, mutation, and population size is able to automate the process of selecting appropriate parameter values and it is able to find good values for these parameters quickly during its run. The ANSGA-II out-performs the original NSGA-II with fixed parameter settings and six variants with one or two adaptable parameters, in terms of finding a diverse set of non-dominated solutions and converging close to the true Pareto-optimal front. The improvement comes with the cost of longer execution time due to overheads of solving the problem and learning good parameter values at the same time. However, the execution time appears to be acceptable on all thirteen benchmark multi-objective problems. There is no definite conclusion on which adaptable parameter among three adaptable parameters (N, p_c, p_m) affects the performance of the ANSGA-II the most or the least. In general, the variants of ANSGA-

II with one or two adaptable parameters among three parameters N, p_c, and p_m out-perform the original NSGA-II on easy problems. On difficult problems, they have mixed performance results. Either adaptable p_c or p_m alone enables the algorithm to converge to the global Pareto-optimal front on the problem ZDT4 while the original NSGA-II is trapped in a local Pareto-optimal front. Other variants (adaptable N and p_c, adaptable N and p_m, adaptable p_c and p_m, and adaptable N alone) also fail to converge to the global Pareto-optimal front on the problem ZDT4. The reason for these variants fail to converge to the true Pareto-optimal front is that there is not enough diversity in the population but the distribution of solutions is good; therefore, the algorithm terminates prematurely. On complex problems with several discontinuous Pareto-optimal fronts such as KUR, adaptable p_c alone or p_m alone fails to obtain non-dominated solutions that cover the entire shape of the Pareto-optimal fronts. The most likely reason is that the algorithm does not have enough time to learn good parameter values given insufficient population size and number of generations. However, the ANSGA-II with adaptable N and p_c, and the ANSGA-II with adaptable N perform better than the original NSGA-II on the problem KUR. Regarding the overhead for adapting parameters and solving the problem at the same time, the variants with adaptable N take longer time than other variants due to overhead of executing multiple populations simultaneously for learning a proper population size. Other variants without adaptable N have execution time comparable to that of required by the NSGA-II. This implies that the ANSGA-II is able to learn good values for crossover and mutation parameters quickly and the cost for adapting population size is expensive.

Additional studies and extensions include: (i) applying and verifying the ANSGA-II on more real-world problems; (ii) applying parameter control techniques used in ANSGA-II to the ε-MOEA, which has a proof of convergence to the true Pareto-optimal set while preserving diversity of the obtained solutions at the same time (Laumanns et al., 2002); (iii) developing more reliable and efficient performance metrics to support parameter control techniques in MOEA.

Appendix A

Test Problems Used in This Study

A suite of benchmark multi-objective test problems used in this study is listed in Table 32 and Table 33. These test problems are selected from a number of significant past studies in MOEA (Deb, Pratap et al., 2002). Brief descriptions of these problems are presented in Section "Results of ANSGA-II with Adaptable N, p_c, p_m, η_c, η_m" of Chapter 4 above. Table 32 lists nine un-constrained test problems that have two objective functions. The problem SCH is borrowed from Schaffer's study (1985); FON from Fonseca & Fleming's study (1998); POL from Poloni's study (1997); KUR from Kursawe's study (1991); and ZDT1, ZDT2, ZDT3, ZDT4, and ZDT6 from Zitzler, Deb, and Thiele's study (2000). The table also shows the number of variables, their bounds, the Pareto-optimal solutions, and the shape of the Pareto-optimal front for each problem.

Table 32: Un-constrained test problems used in this study

Problem	n	Variable bounds	Objective functions to be minimized	Optimal solutions	Comments
SCH	1	[-10^3, 10^3]	$f_1(x) = x^2$ $f_2(x) = (x-2)^2$	$x \in [0, 2]$	convex
FON	3	[-4, 4]	$f_1(x) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right)$ $f_2(x) = 1 - \exp\left(-\sum_{i=1}^{3}\left(x_i - \frac{1}{\sqrt{3}}\right)^2\right)$	$x_1 = x_2 = x_3$ $\in \left[\frac{-1}{\sqrt{3}}, \frac{1}{\sqrt{3}}\right]$	non-convex
POL	2	[$-\pi$,π]	$f_1(x) = \left[1 + (A_1 - B_1)^2 + (A_2 - B_2)^2\right]$ $f_2(x) = [(x_1+3)^2 + (x_2+1)^2]$ $A_1 = 0.5\sin 1 - 2\cos 1 + \sin 2 - 1.5\cos 2$ $A_2 = 1.5\sin 1 - \cos 1 + 2\sin 2 - 0.5\cos 2$		non-convex, disconnected

			$B_1 = 0.5\sin x_1 - 2\cos x_1 + \sin x_2 - 1.5\cos x_2$ $B_2 = 1.5\sin x_1 - \cos x_1 + 2\sin x_2 - 0.5\cos x_2$		
KUR	3	[-5, 5]	$f_1(x) = \sum_{i=1}^{n-1}\left(-10\exp\left(-0.2\sqrt{x_i^2 + x_{i+1}^2}\right)\right)$ $f_2(x) = \sum_{i=1}^{n}\left(\lvert x_i\rvert^{0.8} + 5\sin x_i^3\right)$		non-convex
ZDT1	30	[0, 1]	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{x_1/g(x)}\right]$ $g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$	$x_1 \in [0, 1]$ $x_i = 0$, $i = 2, \ldots n$	convex
ZDT2	30	[0, 1]	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \left(x_1/g(x)\right)^2\right]$ $g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$	$x_1 \in [0, 1]$ $x_i = 0$, $i = 2, \ldots n$	non-convex
ZDT3	30	[0, 1]	$f_1(x) = x_1$ $f_2(x) = g(x)\begin{bmatrix} 1 - \sqrt{x_1/g(x)} - \\ (x_1/g(x))\sin(10\pi x_1) \end{bmatrix}$ $g(x) = 1 + 9\left(\sum_{i=2}^{n} x_i\right)/(n-1)$	$x_1 \in [0, 1]$ $x_i = 0$, $i = 2, \ldots n$	convex, disconnected
ZDT4	10	$x_1 \in [0:1]$ $x_i \in [0:\pi]$ $i = 2, \ldots,$ n	$f_1(x) = x_1$ $f_2(x) = g(x)\left[1 - \sqrt{x_1/g(x)}\right]$ $g(x) = 1 + 10(n-1) +$ $\sum_{i=2}^{n}[x_i^2 - 10\cos(4\pi x_i)]$	$x_1 \in [0, 1]$ $x_i = 0$, $i = 2, \ldots n$	convex
ZDT6	10	[0, 1]	$f_1(x) = 1 - \exp(-4x_1)\sin^6(6\pi x_1)$ $f_2(x) = g(x)\left[1 - \left(f_1(x)/g(x)\right)^2\right]$ $g(x) = 1 + 9\left[\left(\sum_{i=2}^{n} x_i\right)/(n-1)\right]^{0.25}$	$x_1 \in [0, 1]$ $x_i = 0$, $i = 2, \ldots n$	non-convex, non-uniformly spaced

Table 33 lists four constrained test problems. The first three of these problems have two objective functions and the last problem has five objective functions. In the first problem DEB, a part of the unconstrained Pareto-optimal region is not feasible and the resulting constrained Pareto-optimal region is a concatenation of the first constraint boundary and some part of the unconstrained Pareto-optimal region. In the problem SRN, the constrained Pareto-optimal set is a subset of the unconstrained Pareto-optimal set; this problem was used in the original study of NSGA (N. Srinivas & K. Deb, 1994).

The problem TNK, suggested by Tanaka et al., has a discontinuous Pareto-optimal region, entirely falling on the first constraint boundary (Tanaka, Watanabe, Furukawa, & Tantrio, 1995). The last problem WATER has five objective functions and seven constraints (Ray, Kang, & Chye, 2002). The table also shows the number of variables, their bounds, and the constraints for each problem.

Table 33: Constrained test problems used in this study

Problem	n	Variable bounds	Objective functions to be minimized	Constraints
DEB	2	$x_1 \in [0.1:1.0]$ $x_2 \in [0:5]$	$f_1(x) = x_1$ $f_2(x) = (1+x_2)/x_1$	$g_1(x) = x_2 + 9x_1 \geq 6$ $g_2(x) = -x_2 + 9x_1 \geq 1$
SRN	2	$x_i \in [-20:20]$ $i = 1, 2$	$f_1(x) = (x_1-2)^2 + (x_2-1)^2 + 2$ $f_2(x) = 9x_1 - (x_2-1)^2$	$g_1(x) = x_1^2 + x_2^2 \leq 225$ $g_2(x) = x_1 - 3x_2 \leq -10$
TNK	2	$x_i \in [0:\pi]$ $i = 1, 2$	$f_1(x) = x_1$ $f_2(x) = x_2$	$g_1(x) = -x_1^2 - x_2^2 + 1 +$ $0.1\cos(16\arctan x/y) \leq 0$ $g_2(x) = (x_1-0.5)^2 +$ $(x_2-0.5)^2 \leq 0.5$
WATER	3	$0.01 \leq x_1 \leq 0.45$ $0.01 \leq x_2 \leq 0.10$ $0.01 \leq x_3 \leq 0.10$	$f_1(x) = 106780.37(x_2+x_3)$ $+ 61704.67$ $f_2(x) = 3000x_1$ $f_3(x) = \frac{305700 \times 2289x_2}{(0.06 \times 2289)^{0.65}}$ $f_4(x) = 250 \times 2289 \times$ $\exp(-39.75x_2 + 9.9x_3 + 2.74)$ $f_5(x) = 25\left(\frac{1.39}{x_1x_2} + 4940x_3 - 80\right)$	$g_1(x) = \frac{0.00139}{x_1x_2} +$ $4.94x_3 - 0.08 \leq 1$ $g_2(x) = \frac{0.000306}{x_1x_2} +$ $1.082x_3 - 0.0986 < 1$ $g_3(x) = \frac{12.307}{x_1x_2} +$ $49408.24x_3 + 4051.02 \leq 50000$ $g_4(x) = \frac{2.098}{x_1x_2} +$ $8046.33x_3 - 696.71 \leq 16000$ $g_5(x) = \frac{2.138}{x_1x_2} +$ $7883.39x_3 - 705.04 \leq 10000$

				$g_6(x) = \frac{0.417}{x_1 x_2} +$ $1721.26x_3 - 136.54 \le 2000$ $g_7(x) = \frac{0.164}{x_1 x_2} +$ $631.13x_3 - 54.48 \le 550$

Reference List

Abbass, H.A. (2002). *The Self-Adaptive Pareto Differential Evolution Algorithm.* Paper presented at the Congress on Evolutionary Computation (CEC'2002), Piscataway, New Jersey.

Abbass, H.A., Sarkar, R., & Newton, C. (2001). *A Pareto Differential Evolution Approach to Vector Optimization Problems.* Paper presented at the IEEE Congress on Evolutionary Computation, Seoul, Korea.

Bäck, T. (1992). The Interaction of Mutation Rate, Selection & Self-Adaptation within a Genetic Algorithm. In Manner & Manderick (Eds.), *Parallel Problem Solving from Nature* (pp. 85-94): Elsevier Science.

Bäck, T., Eiben, A.E., & van der Vaart, N.A.L. (2000). An empirical study on GAs without parameters. In *Parallel Problem Solving from Nature PPSN V, Lecture Notes in Computer Science* (Vol. 1917, pp. 315-324).

Battle, D.L., & Vose, M.D. (1990). *Isomorphisms of genetic algorithms.* Paper presented at the Foundations of Genetic Algorithms.

Booker, L.B., Fogel, D.B., Whitley, D., & Angeline, P.J. (1997). Recombination. In *The Handbook of Evolutionary Computation (Chapter C3.3, pp. C3.3:1-C3.3:27).* Philadelphia: PA: IOP Publishing Ltd. and Oxford University Press.

Büche, D., Müller, S., & Koumoutsakos, P. (2003). *Self-Adaptation for Multi-objective Evolutionary Algorithms*.

Carson, Y., & Maria, A. (1997). *Simulation optimization: methods and applications.* Paper presented at the Proceedings of the 29th conference on Winter simulation, Atlanta, Georgia, United States: ACM Press.

Cleveland, W.S. (1994). *Elements of Graphing Data*. Murray Hill, NJ: AT & T Bell Laboratories.

Coello, C.A. (2000). An Updated Survey of GA-Based Multiobjective Optimization Techniques. *ACM Computing Surveys, 32*(2).

Coello, C.A. (2001). *A Short Tutorial on Evolutionary Multiobjective Optimization.* Paper presented at the First International Conference on Evolutionary Multi-Criterion Optimization: Springer-Verlag. Lecture Notes in Computer Science.

Coello, C.A. (Ed.). (2003). *Evolutionary Multiobjective Optimization: Current and Future Challenges*: Springer-Verlag.

Coello, C.A. (Ed.). (2005). *Recent Trends in Evolutionary Multiobjective Optimization*: Springer-Verlag.

Coello, C.A., & Christiansen, A.D. (1998). Two New GA-based methods for multiobjective optimization. *Civil Engineering Systems*.

Coello, C.A., & Pulido, G.T. (2001). *A Micro-Genetic Algorithm for Multiobjective Optimization.* Paper presented at the Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization.

Coello, C.A., Pulido, G.T., & Montes, E.M. (Eds.). (2005). *Current and Future Research Trends in Evolutionary Multiobjective Optimization*: Springer-Verlag.

Corne, D.W., & Knowles, J.D. (2003). *No Free Lunch and Free Leftovers Theorems for Multiobjective Optimization Problems.* Paper presented at the Evolutionary Multi-Criterion Optimization. Second International Conference, Faro, Portugal: Springer. Lecture Notes in Computer Science.

Corne, D.W., Knowles, J.D., & Oates, M.J. (2000). *The Pareto Envelope-based Selection Algorithm for Multiobjective Optimization.* Paper presented at the Proceedings of the Parallel Problem Solving from Nature VI Conference.

Czyzak, P., & Jaszkiewicz, A. (1998). Pareto simulated annealing - a metaheuristic for multi-objective combinatorial optimization. *Multi-Criteria Decision Analysis, 7*, 34-47.

Davis, L. (1985). *Applying Adaptive Algorithms to Epistatic Domains.* Paper presented at the Proceedings of the International Joint Conference on Artificial Intelligence.

Davis, L. (Ed.). (1991). *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold.

De Jong, K.A. (1975). *An analysis of the behavior of a class of genetic adaptive systems.* Unpublished Doctoral dissertation, University of Michigan, Ann Arbor.

Deb, K. (1999a). *Genetic Algorithm in Search and Optimization: The Technique and Applications*. Kanpur: Indian Institute of Technology.

Deb, K. (1999b). Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation, 7*(3), 205-230.

Deb, K. (1999c). *Solving goal programming problems using multi-objective genetic algorithms.* Paper presented at the 1999 Congress on Evolutionary Computation, Piscataway, NJ.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*: John Wiley & Sons.

Deb, K. (2003). Multi-objective evolutionary algorithms: introducing bias among Pareto-optimal solutions. *Advances in evolutionary computing: theory and applications*, 263-292.

Deb, K., & Agrawal, S. (1995). Simulated binary crossover for continuous search space. *Complex Systems*(5), 115-148.

Deb, K., & Agrawal, S. (1998). Understanding interactions among genetic algorithm parameters. *In Foundations of Genetic Algorithms 5*, 265-286.

Deb, K., & Goldberg, D.E. (1989). *An investigation of niche and species formation in genetic function optimization.* Paper presented at the Third International Conference on Genetic Algorithms, San Francisco, CA.

Deb, K., & Goyal, M. (1996). A combined genetic adaptive search (GeneAS) for engineering design. *Computer Science and Informatics, 26*(4), 30-45.

Deb, K., & Jain, S. (2002). *Running Performance Metrics for Evolutionary Multi-Objective Optimization.* Paper presented at the Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning (SEAL'02), Nanyang Technical University, Orchid Country Club, Singapore.

Deb, K., Mohan, M., & Mishra, S. (2003). *A Fast Multi-objective Evolutionary Algorithm for Finding Well-Spread Pareto-Optimal Solutions* (KanGAL Report No. 2003003).

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation, 6*(2), 182-197.

Deb, K., Thiele, L., Laumanns, M., & Zitzler, E. (2002). *Scalable Multi-Objective Optimization Test Problems.* Paper presented at the Congress on Evolutionary Computation (CEC'2002), Piscataway, New Jersey: IEEE Service Center.

Devireddy, V., & Reed, P. (2004). *Efficient and Reliable Evolutionary Multiobjective Optimization using e-Dominance Archiving and Adaptive Population Sizing.* Paper presented at the Genetic and Evolutionary Computation Conference (GECCO), Seattle, Washington.

Dias, A.H.F., & Vasconcelos, J.A.d. (2002). Multiobjective Genetic Algorithms Applied to Solve Optimization Problems. *IEEE Transaction on Magnetics, 38*(2), 1133-1136.

Eiben, Á.E., Hinterding, R., & Michalewicz, Z. (1999). Parameter Control in Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation, 3*(2), 124-141.

Farhang-Mehr, A., & Azarm, S. (2002). *Diversity Assessment of Pareto-optimal solution sets: An entropy approach.* Paper presented at the Proceedings of the World Congress on Computational Intelligence.

Flores, S.D., Cegla, B.B., & Cáceres, D.B. (2003). *Telecommunication network design with parallel multi-objective evolutionary algorithms.* Paper presented at the Proceedings of the 2003 IFIP/ACM Latin America conference on Towards a Latin American agenda for network research, La Paz, Bolivia.

Fogarty, T.C. (1989). *Varying the Probability of Mutation in the Genetic Algorithm.* Paper presented at the Proceedings of 3rd International Conference on Genetic Algorithms: Morgan Kaufmann.

Fogel, L.J., Owens, A.J., & Walsh, M.J. (1966). *Artificial Intelligence through Simulated Evolution*: Wiley, New York.

Fonseca, C.M., & Fleming, P.J. (1993). *Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization.* Paper presented at the Proceedings of the Fifth International Conference on Genetic Algorithms, San Mateo, California, 1993: University of Illinois at Urbana-Champaign, Morgan Kaufmann Publishers.

Fonseca, C.M., & Fleming, P.J. (1995). *Multi-objective genetic algorithms made easy: Selection, sharing and mating restriction.* Paper presented at the Proceedings of the First International Conference on Genetic Algorithms in Engineering Systems: Innovations and Application.

Fonseca, C.M., & Fleming, P.J. (1998). Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part II: Application example. *IEEE Transaction on Systems, Man, and Cybernetics: Part A: Systems and Humans*, 38-47.

Fu, M.C. (2001). *Simulation optimization.* Paper presented at the Proceedings of the 33rd conference on Winter simulation, Arlington, Virginia.

Goldberg, D.E. (1985). *Optimal initial population size for binary-coded genetic algorithms* (No. 85001). Tuscaloosa: University of Alabama, The Clearinghouse for Genetic Algorithms.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley.

Goldberg, D.E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. *Foundations of Genetic Algorithms*, 69-63.

Goldberg, D.E., Deb, K., & Clark, J.H. (1992). Genetic algorithms, noise, and the sizing of populations. *Complex Systems, 6*, 333-362.

Gopalakrishnan, G., Minsker, B., & Padera, B. (2001). *Risk Based Corrective Action Design Using Genetic Algorithms.* Paper presented at the Proceedings of the World Water and Environmental Resources Congress, Orlando, FL.

Hajela, P., & Lin, C.Y. (1992). Genetic search strategies in multicriterion optimal design. *Structural Optimization, 4*, 99-107.

Hanne, T. (2000a). Global multi-objective optimization using evolutionary algorithms. *Journal of Heuristics, 6*(3), 347-360.

Hanne, T. (2000b). On the convergence of multi-objective evolutionary algorithms. *European Journal of Operational Research, 117*(3), 553-564.

Harik, G.R., & Lobo, F.G. (1999). *A parameter-less genetic algorithm.* Paper presented at the In Banzhaf, W., Daida, J., Eiben, A. E., Garzon, M. H., Honavar, V., Jakiela, M., & Smith, R. E. (Eds.), GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference. San Francisco, CA: Morgan Kaufmann.

Hart, W.E., & Belew, R.K. (1991). *Optimizing an arbitrary function is hard for the genetic algorithms.* Paper presented at the Proceedings of the Fourth International Conference on Genetic Algorithms.

Hesser, J., & Männer, R. (1991). *Towards an optimal mutation probability for genetic algorithms.* Paper presented at the Parallel Problem Solving from Nature - Proceedings of 1st Workshop, PPSN1, Dortmund, Germany: Springer-Verlag, Berlin, Germany.

Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems* (2nd ed.). Ann Arbor: MI: University of Michigan Press.

Horn, J.r., Nafpliotis, N., & Goldberg, D.E. (1994). *A Niched Pareto Genetic Algorithm for Multiobjective Optimization.* Paper presented at the Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence, Piscataway, New Jersey, June 1994: IEEE Service Center.

Karguta, H., Deb, K., & Goldberg, D.E. (1992). Ordering genetic algorithms and deception. *Parallel Problem Solving from Nature, 2*, 47-56.

Khare, V., Yao, X., & Deb, K. (2002). *Performance Scaling of Multi-objective Evolutionary Algorithms* (No. 2002009). Edgbaston: School of Computer Science, The University of Birmingham.

Khoshgoftar, T.M. (2004). A Multiobjective Module-Order Model for Software Quality Enhancement. *IEEE Transactions on Evolutionary Computation, 8*(6), 593-608.

Knowles, J.D., & Corne, D.W. (1999). *The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Multiobjective Optimization.* Paper presented at the Congress on Evolutionary Computation, Washington, D.C., July 1999: IEEE Service Center.

Knowles, J.D., & Corne, D.W. (2000). Approximating the non-dominated front using the pareto archived evolution strategy. *Evolutionary Computation, 8*(2), 149-172.

Kursawe, F. (1991). A variant of evolution strategies for vector optimization. In *Parallel Problem Solving from Nature* (pp. 193-197): Springer.

Laumanns, M., Rudolph, G., & Schwefel, H.-P. (2001). *Mutation Control and Convergence in Evolutionary Multi-Objective Optimization.* Paper presented at the Proceedings of the 7th International Mendel Conference on Soft Computing (MENDEL 2001), Brno, Czech Republic.

Laumanns, M., Thiele, L., Deb, K., & Zitzler, E. (2002). Combining convergence and diversity in evolutionary multi-objective optimization. *Evolutionary Computation, 10*(3).

Lobo, F.G., & Goldberg, D.E. (2001). *The Parameter-less Genetic Algorithm in Practice* (No. IlliGAL Report No. 2001022). Urbana: University of Illinois at Urbana-Champaign.

Lu, H., & Yen, G. (2002). *Rank-density based multiobjective genetic search.* Paper presented at the Proceedings of the World Congress on Computational Intelligence.

Luger, G.F. (2002). *Artificial Intelligence Structure and Strategies for Complex Problem Solving* (4 ed.). New York: Addison Wesley.

Magoulas, G.D., Eldabi, T., & Paul, R.J. (2002). *Global Search Strategies for Simulation Optimization.* Paper presented at the Proceedings of the 34th conference on Winter simulation: exploring new frontiers, San Diego, California.

Maple, C., Guo, L., & Zhang, J. (2004). *Parallel Genetic Algorithms for Third Generation Mobile Network Planning.* Paper presented at the Parallel Computing in Electrical Engineering, International Conference on (PARELEC'04), Dresden, Germany.

Meisel, W.L. (1993). Tradeoff decision in multiple criteria decision making. In J. L. Cochrane & M. Zeleny (Eds.), *Multiple Criteria Decision Making* (pp. 461-476).

Michalewicz, Z. (1996). *Genetic Algorithms + Data Structures = Evolution Programs (3rd ed.)*: Springer-Verlag, London, UK.

Mitchell, M. (2002). *An Introduction to Genetic Algorithms*: The MIT Press.

Mühlenbein, H. (1992). *How genetic algorithms really work: I. Mutation and Hillclimbing.* Paper presented at the Parallel Problem Solving from Nature - PPSN II: Elsevier Science.

Norris, S.R., & Crossley, W.A. (1998). *Pareto-optimal controller gains generated by a genetic algorithm.* Paper presented at the AIAA 36th Aerospace Sciences Meeting and Exhibit, Reno, Nevada.

Oliver, I.M., Smith, D.J., & Holland, J.R.C. (1987). *A study of permutation crossover operators on the traveling salesman problem.* Paper presented at the Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application, October 1987, Cambridge, MA, USA.

Osyczka, A. (Ed.). (1985). *Multicriteria optimization for engineering design*. New York, NY: Academic Press, Inc.

Pelikan, M., & Lobo, F. (1999). *Parameter-less Genetic Algorithm: A Worst-case Time and Space Complexity Analysis.* Paper presented at the Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2000).

Poloni, C. (1997). Hybrid GA for multi-objective aerodynamic shape optimization. In *Genetic algorithms in engineering and computer science* (pp. 397-414): Wiley.

Radcliffe, N.J. (1991). *Formal analysis and random respectful recombination.* Paper presented at the Proceedings of the Fourth International Conference on Genetic Algorithms.

Rao, S.S. (1987). Game theory approach for multiobjective structural optimization. *Computers and Structures, 25*(1), 119-127.

Ray, T., Kang, T., & Chye, S. (2002). Multiobjective design optimization by an evolutionary algorithm. *Engineering Optimization*.

Rechenberg, I. (1965). *Cybernetic Solution Path of an Experimental Problem*: Ministry of Aviation, Royal Aircraft Establishment (U.K.).

Rogers, J.L. (2000). *A parallel approach to optimum actuator selection with a genetic algorithm.* Paper presented at the AIAA Guidance, Navigation, and Control Conference, Denver, CO.

Ronsenberg, R.S. (1967). *Simulation of genetic populations with biochemical properties.* Unpublished PhD dissertation, University of Michigan.

Rudolph, G. (1998). *On a Multi-Objective Evolutionary Algorithm and Its Convergence to the Pareto Set.* Paper presented at the Proceedings of the 5th IEEE Conference on Evolutionary Computation, Piscataway, New Jersey: IEEE Press.

Rudolph, G. (2001). *Evolutionary search under partially ordered fitness sets.* Paper presented at the Proceedings of the International Symposium on Information Science Innovations in Engineering of Natural and Artificial Intelligent Systems (ISI 2001).

Rudolph, G., & Agapie, A. (2000). *Convergence properties of some multi-objective evolutionary algorithms.* Paper presented at the Proceedings of the 2000 Conference on Evolutionary Computation, Piscataway, NJ: IEEE Press.

Sbalzarini, I.F., Müller, S., & Koumoutsakos, P. (2001). *Microchannel optimization using multiobjective evolution strategies.* Paper presented at the Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization, Zürich, Switzerland.

Schaffer, D.J. (1985). *Multiple objective optimization with vector evaluated genetic algorithms.* Paper presented at the Proceedings of the First International Conference on Genetic Algorithms, Hillsdale, New Jersey.

Schnier, T., Yao, X., & Liu, P. (2001). *Digital Filter Design Using Multiple Pareto Fronts.* Paper presented at the Third NASA/DoD Workshop on Evolvable Hardware, Long Beach, California.

Smith, J.E., & Fogarty, T.C. (1996). *Self Adaptation of Mutation Rates in a Steady State Genetic Algorithm.* Paper presented at the Proceedings of the 1996 IEEE International Conference on Evolutionary Computing: IEEE Press.

Spears, W.M. (1991). *Adapting crossover in a genetic algorithm.* Paper presented at the Proceedings of the 4th International Conference on Genetic Algorithms: Morgan Kaufmann.

Spears, W.M. (1997). Recombination parameters. In T. Back, D. B. Fogel & Z. Michalewicz (Eds.), *The Handbook of Evolutionary Computation (Chapter E1.3, pp. E1.3:1-E1.3:13)*. Philadelphia: PA: IOP Publishing Ltd. and Oxford University Press.

Spears, W.M., & De Jong, K.A. (1991). *An analysis of multi-point crossover.* Paper presented at the Foundations of Genetic Algorithms.

Sridhar, J., & Rajendran, C. (1996). Scheduling in Flowshop and Cellular Manufacturing Systems with Multiple Objectives - A Genetic Algorithm Approach. *Production Planning & Control, 7*(4), 374-382.

Srinivas, M., & Patnaik, L.M. (1994). Genetic Algorithms: A Survey. *IEEE Transaction on Computer*, 17-26.

Srinivas, N., & Deb, K. (1994). Multiobjective optimization using non-dominated sorting in genetic algorithms. *Evolutionary Computation, 2*(2), 221-248.

Srinivas, N., & Deb, K. (1994). Multiple objective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation, 2*(3), 221-248.

Storn, R., & Price, K. (1996). *Minimizing the real functions of the icec'96 contest by differential evolution.* Paper presented at the IEEE Conference on Evolutionary Computation.

Tanaka, M., Watanabe, H., Furukawa, Y., & Tantrio, T. (1995). *GA-based decision support system for multi-criteria, optimization.* Paper presented at the Proceedings of the International Conference on Systems, Man and Cybernetics-2.

Tate, D.M., & Smith, E.A. (1993). *Expected allele coverage and the role of mutation in genetic algorithms.* Paper presented at the Proceedings of the Fifth International Conference on Genetic Algorithms.

Tran, K.D. (2005). *Elitist Non-Dominated Sorting Genetic Algorithm-II (NSGA-II) as a Parameter-less Multi-Objective GA.* Paper presented at the IEEE SoutheastCon 2005, Fort Lauderdale, Florida, USA.

Veldhuizen, D.A.V. (1999). *Multiobjective Evolutionary Algorithms: Classifications, Analysis, and New Innovations.* Unpublished PhD, Air University.

Veldhuizen, D.A.V., & Lamont, G.B. (1998). Evolutionary computation and convergence to a pareto front. In J. R. Koza (Ed.), *Late Breaking Papers at the Genetic Programming 1998 Conference* (pp. 221-228). Sanford, CA: Stanford University Bookstore.

Venugopal, V., & Narendran, T.T. (1992). A genetic algorithm approach to the machine-component grouping problem with multiple objectives. *Computers and Industrial Engineering, 22*(4), 469-480.

Wienke, P.B., Lucasius, C., & Kateman, G. (1992). Multicriteria target optimization of analytical procedures using a genetic algorithm. *Analytical Chimica Acta, 265*(2), 211-225.

Wilson, P.B., & Macleod, M.D. (1993). *Low implementation cost IIR digital filter design using genetic algorithms.* Paper presented at the IEE/IEEE Workshop on Natural Algorithms in Signal Processing, Chelmsford, U.K.

Yang, J.-M., & Hwang, C.-N. (2002). Optimization of Corrugated Bulkhead Forms by Genetic Algorithm. *Journal of Marine Science and Technology, 10*(2), 146-153.

Zebulum, R.S., Pacheco, M.A., & Vellasco, M. (1998). *A multi-objective optimization methodology applied to synthesis of low-power operational amplifiers.* Paper presented at the Proceedings of the XIII International Conference in Microelectronics and Packaging, Curitiba, Brazil.

Zitzler, E. (1999). *Evolutionary algorithms for multiobjective optimization: Methods and applications.* Unpublished Doctoral, Swiss Federal Institute of Technology (ETH), Zurich.

Zitzler, E., Deb, K., & Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation, 8*(2), 173-195.

Zitzler, E., Laumanns, M., & Bleuler, S. (2004). A Tutorial on Evolutionary Multiobjective Optimization. In *Metaheuristics for Multiobjective Optimization* (pp. 3-38). Berlin, Germany: Springer-Verlag.

Zitzler, E., Laumanns, M., & Thiele, L. (2002). SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. In *Evolutionary Methods for Design, Optimization, and Control* (Vol. 16, pp. 19-26). Barcelona, Spain: CIMNE.

Zitzler, E., & Thiele, L. (1998). *An Evolutionary Approach for Multiobjective Optimization: The Strength Pareto Approach* (TIK Report No. 43). Zurich: Computer Engineering and Networks Lab (TIK), Swiss Federal Institute of Technology (ETH).

Zitzler, E., & Thiele, L. (1999). Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transaction Evolutionary Computation, 3*(4), 257-271.