Le, Khoi Nguyen (2011) A study of evolutionary multiobjective algorithms and their application to knapsack and nurse scheduling problems. PhD thesis, University of Nottingham.

A STUDY OF EVOLUTIONARY MULTIOBJECTIVE ALGORITHMS AND THEIR APPLICATION TO KNAPSACK AND NURSE SCHEDULING PROBLEMS

by Khoi Nguyen Le, BSc, MRes

Thesis submitted to the University of Nottingham

for the degree of Doctor in Philosophy

School of Computer Science

March 2011

Contents

List of Figures vi

List of Tables ix

Abstract xi

Acknowledgements xii

1 Introduction 1

1.1 Motivation 1

1.2 Structure of this Thesis 3

1.3 Contribution of this Thesis 4

2 Nurse Scheduling Problems 5

2.1 Nurse Scheduling Problems 6

2.1.1 Basic Terminologies 7

2.1.2 Constraints in Nurse Scheduling 9

2.1.3 Objective Functions 11

2.1.4 Generic Models 11

2.2 Nurse Scheduling Approaches 12

2.2.1 Simulated Annealing 13

2.2.2 Tabu Search 14

2.2.3 Evolutionary Algorithms 15

2.2.4 Other Approaches 18

2.3 Summary 19

3 Evolutionary Multiobjective Optimisation **21**

3.1 Multiobjective Optimisation 23

3.2 Evolutionary Algorithms 27

3.2.1 Overview of Evolutionary Algorithms 28

3.2.2 Multiobjective EAs in the Literature 38

3.3 Performance Assessment 46

3.3.1 The 0/1 Multiple Knapsack Problem 46

3.3.2 Performance Metrics 52

4 Restricted Assortative Mating in EMO algorithms **57**

4.1 Introduction 57

4.2 Mating Schemes for EMOAs 59

4.3 The Assortative Mating Scheme 61

4.3.1 The Experimental Setting 62

4.3.2 Similarity Measurement 62

4.3.3 Static Setting of the Mating Pressure 63

4.3.4 Dynamic Setting of the Mating Pressure 67

4.4 Summary 72

5 Volume Dominance in EMO **73**

5.1 Introduction 74

5.2 Literature Review 75

5.3 Volume Dominance 79

5.3.1 Experimental Design 82

5.3.2 Results and Discussion 85

5.3.3 Summary 94

5.4 Improved Volume Dominance 94

5.4.1 Improved Volume Dominance 96

5.4.2 Experimental Design 100

5.4.3 Results and Discussion 101

5.5 Summary 107

6 Hyper Volume Evolutionary Algorithm 108

6.1 Hyper Volume Evolutionary Algorithm 109

6.1.1 Fitness Assignment 111

6.1.2 Ranking Assignment 113

6.1.3 Environmental Selection 114

6.1.4 Crowding Assignment 115

6.1.5 Generating Offspring 117

6.1.6 Comparison With SMS-EMOA 117

6.2 Comparative Case Study 118

6.2.1 Evolutionary Multiobjective Algorithms 118

6.2.2 Experimental Design 118

6.3 Results and Discussion 121

6.3.1 Comparison to EMO algorithms 121

6.3.2 Comparison to MOEA/D 129

6.3.3 Further Discussion 135

6.4 Summary 136

7 Theoretical Model To Real-world Nurse Scheduling Problems 138

7.1 Introduction 138

7.2 Queens Medical Centre Problem 141

7.2.1 Problem Description 142

7.2.2 Problem Modelling 144

7.2.3 Self-Adaptive Heuristic Decoder 147

7.2.4 Preliminary Investigations 150

7.3 A New Model for QMC Problem 153

7.3.1 Hard Constraint Violation Repaired Heuristics 154

7.3.2 A More Deterministic Heuristic Approach 156

7.4 A More General Approach to Nurse Scheduling Problems - A Proposal 159

7.5 Summary 161

8 Conclusions and Future Work **162**

8.1 Conclusions 162

8.2 Future Work 165

Bibliography **167**

Appendix - List of Publications **183**

List of Figures

3.1 One Point Crossover 33

3.2 Two Point Crossover 33

3.3 Uniform Crossover 33

3.4 Cycle Crossover 34

3.5 Uniform Order-Based Crossover 34

3.6 The choices of the reference point ref_1 (close to Pareto sets) and ref_2 (at the origin) impose different performance assessment of Pareto sets X' and X'' regarding the $\mathcal{S}$-metric. 52

3.7 X' is better than X'' but X' and X'' are similar under the $\mathcal{C}$-metric. 53

4.1 Performance of various EMO algorithms on the multiple 0/1 knapsack problem with respect to percentage of the complement of the $\mathcal{S}$ metric. 65

4.2 Results of SEAMO2(RM) on the 2-knapsack and 750 items problem for six values of σ_{mating}. 66

4.3 Performance of various EMO algorithms on the multiple 0/1 knapsack problem with respect to percentage of the complement of the $\mathcal{S}$ metric. 68

4.4 Results comparing NSGA2, SPEA2, SEAMO2, SE2I, SE2S and SE2D on the 2-knapsack problem with 750 items. 70

4.5 Results comparing SE2S and SE2D on the 2-knapsack problem with 750 items. 71

5.1 Volume Dominance - A Form of Relaxed Dominance 80

5.2 The performance of Pareto dominance and volume dominance on SEAMO2 for 4-objective knapsack problem on the reciprocal of the $\mathcal{S}$-metric. 87

5.3 The distribution of objective values obtained when using conventional Pareto dominance and volume dominance on SEAMO2 for 4-objective knapsack problem. 88

5.4 The performance of Pareto dominance and volume dominance on SPEA2 for 4-objective knapsack problem on the reciprocal of the $\mathcal{S}$-metric. 90

5.5 The performance of Pareto dominance and volume dominance on NSGA2 for 4-objective knapsack problem on the reciprocal of the $\mathcal{S}$-metric. 91

5.6 The distribution of objective values obtained when using conventional Pareto dominance and volume dominance on SPEA2 for 4-objective knapsack problem 92

5.7 The distribution of objective values obtained when using conventional Pareto dominance and volume dominance on NSGA2 for 4-objective knapsack problem 93

5.8 *Volume dominance.* In the case of the convex front, both points should be regarded as equally good instead of X being evaluated as better than X* 95

5.9 Volume Dominance - Reference Point. 97

5.10 The fairly representative Pareto front of the population with respect to the individual $\vec{x}$. 98

5.11 Performance of Pareto dominance, previous *volume dominance* and new improved *volume dominance* on SEAMO2, SPEA2, NSGA2 for 2-, 3-, and 4- objective knapsack problems with 750 items on the reciprocal of the $\mathcal{S}$-metric. 102

5.12 The combined non-dominated front obtained from 30 independent runs for 2-objective knapsack problem with 750 items. 106

6.1 The current representative Pareto front of the population with respect to the individual $\vec{x}$ 109

6.2 The hypervolume $\mathcal{S}$-metric (permutation encoding) 124

6.3 The hypervolume $\mathcal{S}$-metric (binary encoding) 124

6.4 The hypervolume $\mathcal{S}$-metric (te binary encoding) 125

6.5 The hypervolume $\mathcal{S}$-metric (ws binary encoding) 125

6.6 The hypervolume $\mathcal{S}$-metric (permutation encoding) 130

6.7 The hypervolume $\mathcal{S}$-metric (binary encoding) 130

6.8 The hypervolume $\mathcal{S}$-metric (te binary encoding) 131

6.9 The hypervolume $\mathcal{S}$-metric (ws binary encoding) 131

7.1 Constructed Schedule, Preference Schedule and Permutation List . . 145

7.2 The decoding process. 148

7.3 Repair the violation of the *Succession* constraint. 149

7.4 Repair the violation of the *MinDaysOn* constraint. 149

7.5 Performance, based on the $\mathcal{S}$-metric, of EMO algorithms incorporated the heuristic decoder on the QMC model proposed in [81]. . . 152

7.6 Performance, based on the $\mathcal{S}$-metric, of EMO algorithms incorporated the heuristic decoder without repairing *MinDaysOn* hard constraint on the QMC model proposed in [81] using objective function replacement. 155

7.7 The decoding process using single permutation list of shifts. 157

7.8 Performance, based on the $\mathcal{S}$-metric, of EMO algorithms on the new QMC model using single permutation and a more deterministic heuristic decoder 158

List of Tables

4.1 Average values (standard deviation) of *coverage of two sets* $\mathcal{C}(A \succeq B)$. 70

5.1 The performance of Pareto dominance and volume dominance on SEAMO2 for 4-objective knapsack problem on the $\mathcal{C}(A \succ B)$ metric. 87

5.2 The performance of Pareto dominance and volume dominance on SPEA2 for 4-objective knapsack problem on the $\mathcal{C}(A \succ B)$ metric. . 90

5.3 The performance of Pareto dominance and volume dominance on NSGA2 for 4-objective knapsack problem on the $\mathcal{C}(A \succ B)$ metric. . 91

5.4 Average size (standard deviation) of the nondominated set. 103

5.5 Average size (standard deviation) of the cluster metrics CL_{μ}. 104

5.6 Average distance (standard deviation) from the nondominated set to the approximation of the true Pareto Front. 105

6.1 Parameter Setting for The Multiple 0/1 Knapsack Problem 119

6.2 Generational Distance (permutation encoding) 126

6.3 Generational Distance (binary encoding) 126

6.4 Generational Distance (te binary encoding) 126

6.5 Generational Distance (ws binary encoding) 126

6.6 Inverted Generational Distance (permutation encoding) 127

6.7 Inverted Generational Distance (binary encoding) 127

6.8 Inverted Generational Distance (te binary encoding) 127

6.9 Inverted Generational Distance (ws binary encoding) 127

6.10 Computational Time in seconds (permutation encoding) 128

6.11 Computational Time in seconds (binary encoding) 128

6.12 Computational Time in seconds (te binary encoding) 128

6.13 Computational Time in seconds (ws binary encoding) 128

6.14 Generational Distance (permutation encoding) 132

6.15 Generational Distance (binary encoding) 132

6.16 Generational Distance (te binary encoding) 132

6.17 Generational Distance (ws binary encoding) 132

6.18 Inverted Generational Distance (permutation encoding) 133

6.19 Inverted Generational Distance (binary encoding) 133

6.20 Inverted Generational Distance (te binary encoding) 133

6.21 Inverted Generational Distance (ws binary encoding) 133

6.22 Computational Time in seconds (permutation encoding) 134

6.23 Computational Time in seconds (binary encoding) 134

6.24 Computational Time in seconds (te binary encoding) 134

6.25 Computational Time in seconds (ws binary encoding) 134

6.26 Computational Time (in seconds) for The Multiple 0/1 Knapsack Problem 136

7.1 *Coverage* demand of nurses in each shift. 144

7.2 Measurement of the distribution of nurses (the objective function 4). 147

Abstract

Evolutionary algorithms (EAs) based on the concept of Pareto dominance seem the most suitable technique for multiobjective optimisation. In multiobjective optimisation, several criteria (usually conflicting) need to be taken into consideration simultaneously to assess a quality of a solution. Instead of finding a single solution, a set of trade-off or compromise solutions that represents a good approximation to the Pareto optimal set is often required. This thesis presents an investigation on evolutionary algorithms within the framework of multiobjective optimisation. This addresses a number of key issues in evolutionary multiobjective optimisation. Also, a new evolutionary multiobjective (EMO) algorithm is proposed. Firstly, this new EMO algorithm is applied to solve the multiple 0/1 knapsack problem (a well-known benchmark multiobjective combinatorial optimisation problem) producing competitive results when compared to other state-of-the-art MOEAs.

Secondly, this thesis also investigates the application of general EMO algorithms to solve real-world nurse scheduling problems. One of the challenges in solving real-world nurse scheduling problems is that these problems are highly constrained and specific-problem heuristics are normally required to handle these constraints. These heuristics have considerable influence on the search which could override the effect that general EMO algorithms could have in the solution process when applied to this type of problems. This thesis outlines a proposal for a general approach to model the nurse scheduling problems without the requirement of problem-specific heuristics so that general EMO algorithms could be applied. This would also help to assess the problems and the performance of general EMO algorithms more fairly.

Acknowledgements

I would like to thank my supervisor Dr. Dario Landa-Silva for all the support and guidance which he had given me throughout my doctoral study. Also, thanks to Dr. Rong Qu who gave me valuable comment and advice during the second year review of my doctoral degree. I also thank the School of Computer Science, University of Nottingham for providing excellent facilities for my research and the Graduate School of the University of Nottingham for their research training programme.

This doctoral programme would have not been possible without the financial support of the Overseas Research Students Awards Scheme (ORSAS) and the School of Computer Science, University of Nottingham.

I would like to thank my family and friends for their encouragement.

Finally, thanks to all members of ASAP group, School of Computer Science for their support and excellent working environment.

Chapter 1

Introduction

1.1 Motivation

Many real-world optimisation problems are multiobjective by nature due to several criteria associated with them. Solving multiobjective optimisation problems by confronting these possibly conflicting criteria is one of the main challenges to a decision-maker. A suitable solution, which represents a good compromise between these criteria, is required. However, it is quite difficult to find such a solution due to the several criteria that need to be taken into consideration simultaneously. There are three approaches for selecting this good compromise solution which are categorised based on the process of handling the search and the decision making. These three approaches are a-priori preference articulation (decision making before search), a-posteriori preference articulation (search before decision making) and interactive preference articulation (interactive search and decision making) [33]. Within the scope of this thesis, the a-posteriori approach, which constructs a set of alternative solutions for the decision maker to select a suitable solution, will be discussed in detail. The challenge to this approach is that, without any preference to criteria, this set of solutions should be both well-converged and well-distributed.

This set is known as the Pareto front in Pareto-based multiobjective optimisation. Solutions in this set are said to be non-dominated, i.e. none of the solutions is better than others in the set.

Evolutionary algorithms (EAs) are capable of generating multiple promising solutions in a single run and evolving a population of solutions towards the Pareto front. These properties make EAs especially adequate to deal with Pareto-based multiobjective optimisation problems (MOPs). In EAs, a population is maintained and evolved throughout the search process. Solutions in the population could survive and pass their characteristics to the next generations. This process is an analogy to biological evolution [37].

Over recent years, multiobjective optimisation problems and the application of evolutionary approaches to deal with these problems have received increasing attention from the research community. Some of the most interesting and difficult MOPs are real-world nurse scheduling problems in the field of personnel scheduling. There are a large number of constraints associated to real-world nurse scheduling problems. It is non-trivial to satisfy these constraints. It normally requires a particular heuristic or metaheuristic to handle constraints of a given problem. Then, evolutionary approaches could be applied to optimise these solutions. A conventional approach to solve nurse scheduling problems using evolutionary algorithms is to employ a simple evolutionary algorithm incorporating problem-specific heuristics/metaheuristics. This approach could generate high quality solutions. However, it is difficult to employ these heuristics/metaheuristics for other problems. There is interest in developing more general heuristics/metaheuristics which could be incorporated into general, strong performing EAs to solve nurse scheduling problems.

The work reported in this thesis presents an investigation on general evolutionary multiobjective optimisation (EMO) algorithms and the application of EMO algorithms in solving nurse scheduling problems.

1.2 Structure of this Thesis

The remainder of this thesis is organised as follows.

Chapter 2 provides the basic terminology for nurse scheduling problems. It also discusses common constraints and generic models related to nurse scheduling problems. A brief review on different approaches to solve nurse scheduling problems, which concentrates on evolutionary approaches, is presented.

Chapter 3 provides the basic concepts of multiobjective optimisation problems and evolutionary multiobjective optimisation. Key issues in designing EMO algorithms are discussed in this chapter. Some of important and recent EMO algorithms are described. Chapter 3 also presents a set of performance metrics to assess EMO algorithms. A benchmark problem for evolutionary multiobjective optimisation, the multiple 0/1 knapsack problem, is presented and reviewed.

Chapter 4 proposes an adaptive assortative mating scheme in the decision space which could be incorporated into EMO algorithms. This mating scheme adapts the mating pressure as the search progresses.

Chapter 5 provides a brief review on relaxed Pareto dominance. It proposes a new form of relaxed Pareto dominance which is based on the concept of the hypervolume of individuals in the objective space.

Chapter 6 further explores the ideas of volume dominance proposed in chapter 5. A new population-based EMO algorithm, called hyper volume evolutionary algorithm (HVEA), is proposed and compared to other state-of-the-art EMO algorithms.

Chapter 7 reviews a real-world nurse scheduling problem, the QMC problem [81]. This chapter investigates and proposes a new model for the QMC problem which could be solved by general EMO algorithms. Based on this study of the

new model for the QMC problem, a more general approach to nurse scheduling problems is outlined.

Chapter 8 briefly summarises the work investigated in this thesis. It also suggests future work which was drawn up based on the study of this thesis.

1.3 Contribution of this Thesis

The contributions of this thesis are as follows:

- A review of common solution representation and constraint handling techniques for the multiple 0/1 knapsack problem, a well-known benchmark problem in multiobjective optimisation.
- A new form of relaxed Pareto dominance, named *volume dominance* is proposed. The concept of *volume dominance* is based on comparing the dominated volume of individuals to establish superiority between solutions in the population. It is shown that this new form of relaxed Pareto dominance incorporated in different EMO algorithms is more robust than the conventional Pareto dominance which allows EMO algorithms to obtain more consistent performance.
- A new EMO algorithm, hyper volume evolutionary algorithm (HVEA), which is based on the concept of *volume dominance*, is presented. HVEA exploits the principle of *volume dominance* to estimate individual fitness, rank and crowding around an individual. Results show that HVEA outperforms or remains competitive to other state-of-the-art EMO algorithms.
- A more general EMO approach to solve nurse rostering problems based on exploiting constraint handling techniques for the multiple 0/1 knapsack problem.

Chapter 2

Nurse Scheduling Problems

Personnel scheduling or *personnel rostering* refers to the construction of rosters for personnel (or staff) in an organisation over a scheduling period in such a way that the organisation fulfils the demand for its goods or services [51] while also adhering to existing certain work regulations. *Personnel scheduling* problems exist in almost all working environments. It is particularly a difficult and interesting problem for organisations which have a large pool of personnel with multiple skills and work around the clock [7], such as fire brigade and rescue and health care services. High quality schedules help to provide better services, improve overall job satisfaction and better utilise the available workforce [81]. The process of constructing personnel rosters consists of two parts [51]. The first part is to determine the number of personnel, the set of personnel skills (or qualifications), personnel preferences, workplace regulations, service demand, the layout of the work timetable, constraints and other criteria (which vary amongst organisations). This first step could be referred to as the problem construction process. Then, the second step is to allocate suitable staff members to shifts in order to meet the demand of services a different times while attempting to satisfy workplace regulations and personnel preferences [11, 14, 50, 51]. Another possible need within the allocation step is to assign specific job requirements (or duties) to individual personnel members

in each shift. It is often difficult to find feasible schedules that satisfy all hard constraints, such as work regulations, employee preferences and job requirements. It is extremely difficult to find good-quality feasible schedules to these type of highly constrained and complex problems and even more difficult to find optimal solutions [51]. Within the personnel scheduling field, *nurse scheduling* and *airline crew scheduling* are the two sub-fields which have attracted much attention from researchers [51]. This chapter pays attention to the process of allocating staff members to shifts in *nurse scheduling*. The chapter will provide the basic terminologies and discuss the main approaches to tackle nurse scheduling problems proposed in the literature.

2.1 Nurse Scheduling Problems

Nurse scheduling is particularly difficult because of its nature that demands working around the clock in hospitals. Nurses' well being and job satisfaction are affected by irregular shift work [24]. It is usually desired to meet nurse requests while confronting work regulation and other constraints during the personnel scheduling process. Producing good quality nurse schedules has a great impact on the quality of healthcare service, improving overall job satisfaction and making more efficient use of the workforce [30, 81]. Many healthcare institutions use software to support the construction of nurse schedules but in many other cases this is still done manually [24]. For problems of considerable size, the non-automated construction of nurse schedules is time consuming, difficult and prone to errors. As Burke et al. note, "the automatic generation of high quality nurse schedules can lead to improvements in hospital resource efficiency, staff and patient safety, staff and patient satisfaction and administrative workload" [24]. This section provides basic terminology in nurse scheduling. It also discusses work regulations and other constraints in nurse scheduling which are mostly found in the literature.

2.1.1 Basic Terminologies

2.1.1.1 Personnel

Within the hospital environment staff members are organised into *teams of nurses*, also known as *wards*. Usually, each ward performs a set of fixed activities at a settled location [23]. Each nurse has some certain attributes as follows:

Work regulations. This is the agreement and job description such as full-time/part-time basis, permanent/temporary employee, working shifts etc. It is common to have a personalised agreement for each nurse [23].

Qualifications and Skills. Personnel can be categorised based on a number of factors such as qualifications, skills, experience and responsibility [7, 23, 24]. In some cases, gender, nationality and personality are also of concern [7]. Based on their characteristics as well as particular nurse scheduling problems, *qualifications* and *skills* could be further classified such as disjoint, hierarchical, alternative. Disjoint skills set represents skills defined by a single criterion. Hierarchical skills sets means that nurses with higher qualifications could substitute those with lower. Alternative skills sets means that nurses with more experience could substitute higher skill nurses.

2.1.1.2 Planning Period, Shifts and Scheduling

Planning Period. The planning period defines the time horizon over which personnel is scheduled. The typical planning period in nurse scheduling is 4 weeks (28 days).

Shift Type. Shift types (or shifts for short) are periods of work which are usually well defined by starting and ending times within a 24 hour period. Burke et al. [24] pointed out that many nurse scheduling problems employ three common shifts:

early, late and night. There are also other shift types such as day, short early, short late [22] an "unusual" shift [81]. The starting and ending times usually define the length of the shift which will have implication of certain work regulations. These shift types are also referred to as working shifts.

Day-on and Day-off. A day, for which a nurse is allocated one of the above shift types, is known as a working day or *day-on.* While a *day-off* is a day in which the nurse is not allocated a working shift.

Individual preferences. It is often the case that a nurse requests a day-off, annual leave or specific working shifts for some days of the planning period.

Shift sequence/Shift pattern. A shift sequence [17] is a set of shift types on consecutive days, one shift a day. Shift sequences often have different lengths. A shift pattern [3] is often a fixed length set consisting of working shifts and non-working shifts. It is noted that in this case, a day-off is considered as a non-working shift.

Schedule. The definition of a schedule (or a *nurse schedule*) is mainly given by the shift structures employed by nurse scheduling problems. A schedule is an ordered list of working shifts and days-off, or an ordered list of shift sequences and day-off periods, or an ordered list of one or more shift patterns. Then, the length of the schedule is the length of this ordered list which must be the same as the planning period. Nurse scheduling is often described as a non-cyclical approach. Non-cyclical approaches in nurse scheduling provide a greater degree of flexibility and ability to handle individual preferences, unpredicted demand and disruption due to nurse absences [24].

Ward schedule. The ward schedule consists of all nurse schedules combined for the same planning period.

Coverage demand. The coverage demand (or coverage for short) indicates the

required number of nurses with specific qualification for each shift on a particular day during the planning period.

2.1.2 Constraints in Nurse Scheduling

Nurse scheduling problems are very difficult because many constraints are imposed on the problems. Different problems often possess different sets of constraints. Even if employing the same set of constraints, the type assigned to the constraint (hard or soft) could be different. Constraints are broadly classified into two types: hard constraints and soft constraints. Hard constraints are those that must be satisfied in order to make schedules feasible. Soft constraints are desirable but do not have to be satisfied. Cheang et al. [30] presented a good summary of nurse scheduling constraints. The most common nurse scheduling constraints in the literature are described below. These constraints are related to and categorised accordingly to *coverage demand*, *work regulations* and *individual preferences.*

2.1.2.1 Work Regulation Constraints

Most constraints in nurse scheduling fall into this sub-category because work regulations define many rules which should be obeyed during the scheduling process. The main types of work regulation constraints are below:

- *Working hours* define the maximum/minimum hours that a nurse works over a period of time (a week or a fortnight).
- *Consecutive working shifts/days* define the maximum/minimum number of shifts/days that a nurse works in a row. Maximum consecutive working shifts/days allow regular breaks in the schedule of a nurse.
- *Shift Patterns* define illegal and/or undesired patterns of shift types.

- *Shift Assignments* define the maximum/minimum number of shifts that a nurse works in the planning period.

- *Working Weekends* define constraints related to weekend work. For example maximum/minimum number of weekends that nurses work during the planning period or the requirement that nurses should work both days of a weekend.

- *Break periods* define maximum/minimum amounts of time breaks between consecutive working shift patterns.

2.1.2.2 Coverage Demand Constraints

Coverage demand constraints impose an adequate level of staff to meet the patient demands which usually define the required number of nurses during the planning period. In more detail, coverage demand constraints indicate the number of nurses with certain qualifications and skills for any shift on any day over the whole planning period. The required number of nurses could be a maximum, minimum or exact number.

2.1.2.3 Individual Preference Constraints

According to Ernst et al., the tendency in the modern workplace is to focus on individuals rather than on teams and hence, personnel schedules should cater to individual preferences [50]. This is particularly true in nurse scheduling because it is common that each nurse indicates his/her preference schedule. This allows nurses to get involved in the scheduling process. This might help to increase the nurse satisfaction level by attempting to meet the individual preference constraints. This could then lead to an increase in the quality of service offered to patient.

2.1.3 Objective Functions

Objective functions are designed to assess the quality of schedules. Depending on the model used to represent schedules, different approaches could be employed to evaluate objective functions. It is common that objective functions are related to constraints in the model and hence objective functions could measure the violations on constraints or the cost of constraint violation.

2.1.4 Generic Models

Cheang et al. [30] pointed out that there are three common models for the nurse scheduling problem: a nurse-day view, a nurse-time slot view and a nurse-shift pattern view.

1. A *nurse-day* view usually represents a ward schedule as a two-dimensional schedule. One dimension represents the set of days over the planning period and the other dimension represents the nurse schedule. Each variable in the structure represents a shift type (or a day-off) that a nurse works on a particular day. The two-dimensional structure could be extended to consider the shift types as the third dimension. These are known as binary models, where each variable takes a value or either 0 or 1 (with 1 indicating that a nurse works a shift on a day).

2. A *nurse-time slot* view is a variant of the *nurse-day* day. Each decision variable represents the assignment of a nurse to a time period over the planning period. A nurse either receives an assignment or not.

3. A *nurse-shift pattern* view pre-defines a set of shift patterns which are allocated to nurses. Each decision variable represents a shift pattern assigned to a nurse. Usually shift patterns, which are constructed and evaluated before the search, often satisfy certain constraint types. An extension of the

nurse-shift pattern view is the *nurse-shift sequence* view. In the *nurse-shift sequence* view, high quality sequences of shifts are constructed during the search then assigned to nurses.

2.2 Nurse Scheduling Approaches

This section discusses recent and important approaches to nurse scheduling problems in the literature. It pays attention to metaheuristic approaches, especially evolutionary approaches. More comprehensive reviews of nurse scheduling approaches could be found in [24, 30] and more general personnel scheduling approaches in [50, 51].

One of the first methods used for solving nurse scheduling problems (dated back to the 1970s) is mathematical programming [1, 92, 114, 116, 117]. These approaches usually attempt to optimise an objective function defined by the evaluation of violations of certain constraints. It is possible that mathematical programming could find optimal solutions but only to small size problems. However, practical nurse scheduling problems are far more complex and difficult, and mathematical programming maybe inappropriate and impractical to apply. There are only a few papers describing mathematical programming approaches for practical nurse scheduling problems [24].

During the 1980s, goal programming approaches to solve nurse scheduling attracted some attention from researchers. In goal programming, a desired level is defined for each criterion to be achieved. The priorities between criteria are also determined before search. Goal programming then attempts to find the closest solution to each of the criterion targets following their priorities. Mathematical programming is deployed using the principle of goal programming to find solutions [6, 54, 100]. The latest research investigates metaheuristics within a multi-

objective framework [11, 21, 70].

During the 1980s, artificial intelligence techniques were introduced for nurse scheduling problems and until today this area still attracts considerable attention from researchers. These techniques include constraint programming [87, 89, 97] and case-based reasoning [9, 8, 10, 103, 109]. In constraint programming, variables are assigned values in finite domains in order to satisfy all constraints imposed on those variables. Case-based reasoning attempt to solve new problems by exploiting information of solutions to previous similar problems. Case-based reasoning has the ability to learn new knowledge by judging new experience containing information that might help to solve new problems [7].

Over the last two decades, metaheuristic approaches to solve nurse scheduling problems have drawn great attention from researchers. Those metaheuristics include simulated annealing, tabu search, evolutionary algorithm, etc..

2.2.1 Simulated Annealing

Simulated annealing approaches mimic the physical process of annealing in which the quality of solid material is improved by heating up the material then letting it cool down. The simulated annealing algorithm exploits this principle to help the search to escape local optimum and continue searching for the global optimum. A function, which controls the cooling process, indicates the probability that low quality solutions are accepted during the search. This probability is set to high value at the start of the search which allows poor quality solutions to be accepted and hence to easily escape from local optimum. Throughout the search, the probability is gradually reduced, hence it is more difficult to escape from the local optimum. The main purpose is to search in the neighbourhood of current local optimum rather than jump to a new local optimum as it is expected that the global optimum or near global optimum is eventually reached towards the end of

the search.

Burke et al. [28] presented a simulated annealing approach for a multiobjective nurse scheduling problem. They investigated two options to satisfy soft constraints: a weighted-sum evaluation function emphasizing individual preferences, and a domination-based evaluation function encouraging the diversification of the population. Legal shift patterns are assigned to each nurse schedule to satisfy shift-related hard constraints. Then, an adaptive heurictic is used to find a solution by assigning one of the available shift patterns to each nurse. Finally, violations of coverage demands are repaired by modifying under-/over-covered shifts to make solutions feasible.

2.2.2 Tabu Search

Tabu search approaches use a list to hold solutions (or their attributes) that have been visited recently. It is preferred that solutions in this list are not visited again to avoid getting stuck in local optima.

Burke et al.[25] presented tabu search as the drive-force within a hyper-heuristic to solve nurse scheduling problems. Typically, a hyper-heuristic chooses which heuristics and meta-heuristics are attempted to solve the problem based on the current performance of each heuristic.

Burke et al. [22] presented an efficient and effective hybrid tabu search approach for nurse scheduling. An initial feasible solution is generated that satisfy all hard constraints. The hybrid tabu search never violates hard constraints while searching solutions that satisfy as many soft constraint as possible. The hybrid tabu search tries to solve complete weekend constraints first. It then improves the worst nurse schedule by switching part of this schedule with another nurse schedule. The best possible switch is chosen but there is no guarantee on the overall improvement for

every nurse schedule. Finally, their algorithm performs a major step, *greedy shuffling*, which searches all possible switches between parts of two nurses schedules. This step is very time-consuming. The model employs a time interval representation, rather than a shift type representation, which enables shifts to be split and combined. This approach provides greater choice of shift work and part time work which could be closer to real-world nurse scheduling problems.

2.2.3 Evolutionary Algorithms

Evolutionary algorithms are based on the analogy to biological evolution. They maintain a population of solutions. Solutions in the populations are recombined and mutated to form new solutions. The population is evolved by selecting the best solutions from the current population of solutions and also selecting newly generated solutions for the next generation. It is common that in solving nurse scheduling problems, evolutionary algorithms employ some local search techniques such as a mutation process of solutions. The purpose is to not only improve the solution quality but also to maintain the solution feasibility which is one of the main challenges in nurse scheduling.

Aickelin and Dowsland [3] pointed out that the classical generic algorithm is not suitable to handle the conflicts between objectives and constraints which usually occur in nurse scheduling problems. They suggest that problem-specific knowledge should be incorporated to overcome this issue. They proposed three types of problem specific knowledge: co-operating subproblems, incentives, local search/repair. Co-operating subproblems involves decomposing the problem into easier problems based on grades of nurses. The population of the problem and the subpopulations of the subproblems are co-evolved. Incentives are used to penalise solutions with unbalanced shift coverage. In local search/repair, a simple hill-climber cycles through and accepts nurses' shift patterns that improve fitness. The hill-climber is

also there to improve the preferences of feasible solutions. One problem with this approach is that the shift patterns are pre-defined and limited.

Aickelin and Dowsland [4] later proposed an indirect genetic algorithm using an indirect coding and heuristic decoder to the same problem. The heuristic decoder constructs feasible solutions from good quality permutations of nurses which are identified by GA operators. This approach avoids the issue in dealing with problem specific constraints and infeasibility.

Jan et al. [69] proposed a population-less cooperative genetic algorithm approach for a multiobjective nurse scheduling problem. A new solution is produced by applying two-point crossover on two nurse schedules of the same solution. This could be referred to as a mutation on a single solution, hence there is no crossover between solutions. Two objectives in the problem which are subject to minimisation are: minimise the soft constraint violation penalty and minimise the variance on the nurse schedule penalties.

Cai and Li [29] proposed a genetic algorithm for a nurse scheduling problem with three objectives: minimise personnel cost in satisfying coverage requirements, maximise staff surplus, and minimise the variation of the staff surplus. Individual preference is not considered. A crossover operator aims to maintain the diversity and attempts to maximise feasibility of offspring. The most violated constraint is repeatedly repaired by assigning an additional shift.

Kawanaka et al. [74] proposed a genetic algorithm approach for nurse scheduling with various constraints. Crossover operators are applied to create new solutions. Infeasible solutions are then repaired by swapping shifts but trying to maintain parents' characteristic. Hard constraints employed by this model are minimum qualification coverage and other work regulations. Individual preferences, shift patterns and the balance of shifts are soft constraints.

Inoue et al. [64] proposed an interactive approach using an evolutionary algo-

rithm for nurse scheduling. In their approach, users can modify solutions in the population to improve the quality of solutions regarding user's preference. The fitness of a solution measures the violations of coverage demands, illegal shift patterns and individual preferences.

Özcan [99] proposed a memetic approach for a nurse scheduling problem by exploiting a number of mutation, crossover and hill-climbing operators. There are a limited number of constraints. The problem has two types of shifts being assigned over a two week period. The hill-climbing operator changes shift assignments to repair violations on constraints one at a time.

Moz and Pato [94] investigated a genetic algorithm approach for their nurse rescheduling problem which occurs when nurses cannot work in shifts that were previously assigned. The current schedule must be rebuilt if there is no provision for nurse absence. The challenge is that besides satisfying constraints, the new schedule must be as similar as possible to the current one. The approach studied several specific encodings and operators and constructive heuristics for sequencing problems applied to the nurse rescheduling problem. The fitness of individuals is defined based on the similarity between the rebuilt schedule and the current schedule.

Puente et al. [104] present a genetic algorithm for the medical staff scheduling problem. The initial population of feasible solutions is produced by an ad-hoc heuristic schedule builder that satisfies hard constraints. Solutions are generated by a specific crossover operator, which exchanges whole work weeks. Infeasible solutions are then repaired. Individual fitness is based on the weighted sum approach based on satisfying soft constraints.

Landa-Silva and Le [81] proposed a simple evolutionary algorithm with self adaptation for a multiobjective nurse scheduling problem. The problem tackled covers a quite balanced set of hard constraints and soft constraints. New solu-

tions are produced based on crossover of a pair of nurse schedules for the same nurse from two different solutions. Instead of using mutation, Landa-Silva and Le designed a self adaptive decoder which iteratively repairs violations of hard constraints during the construction of nurse schedules. A regeneration strategy, which introduces newly generated solutions to the population to prevent stagnation, is also employed. They set a target of individual preference to be satisfied and other three objectives are: the nurse qualification coverage demand, the variance of coverage demand amongst shifts and work regulations.

Burke et al. [19] proposed a scatter search approach. The idea behind scatter search is to replace stochastic reproduction operators by systematic and strategically designed rules [19]. In scatter search, a new solution is selected for the next generation depending not only on the solution quality but also its contribution to population's diversity. Their approach aims to maintain the highest quality and also a diverse population. In this approach, solution similarity is measured by looking at the number of days in which identical shifts are assigned for each nurse schedule. Then, to obtain a diverse population of solutions, solutions are constructed by assigning a shift to a nurse who has been assigned that shift on a particular day the least number of times in all other solutions in the population. Neighbourhood searches further improve the solution by moving and swapping shifts between nurses.

2.2.4 Other Approaches

Burke et al. [18] proposed a hybrid algorithm which combines a memetic algorithm and tabu search for the nurse scheduling problem [20]. The memetic algorithm employs a steepest decent improvement heuristic as local search. Burke et al. reported that the hybrid approach is better than both tabu search and their memetic algorithms alone. They also report that memetic approaches are more robust than

tabu search but slower.

Aickelin et al. [2] proposed an evolutionary squeaky wheel optimization that incorporates evolutionary selection and mutation operators. This approach is a cycle of analysis, selection, mutation, prioritisation and construction. The analysis step identifies underperforming solution components which are discarded by the selection step based on the fitness of these components. The mutation step continues randomly discarding some components. Incomplete solutions are then repaired by the prioritisation and construction steps. It is argued that improvements in this approach are obtained by selective solution disruption, iterative improvement and constructive repair.

Brucker et al. [17] proposed an adaptive constructive method which is based on a decomposition approach. In this method, high quality shift sequences are constructed during the search, different from [3, 4] where shift patterns are predefined. In the next step, nurse schedules and ward schedules are built based on the shift sequences. During this step constraint violations are considered. To improve the quality of nurse schedules, a greedy local search is also employed during and after the construction of nurse schedules.

2.3 Summary

This chapter presented the basic terminology, concepts and models for the nurse scheduling problem. The chapter also presented a brief description of constraints which are found most often in the literature as well as in real-world nurse scheduling scenarios. However, the chapter does not suggest how to handle constraints. The reason for this is that different models using different solution approaches are likely to use different constraint handlers. It is quite difficult to develop a generic approach for constraint satisfaction.

This chapter also summarised some approaches to nurse scheduling in the literature, but paid more attention to evolutionary approaches. Evolutionary approaches are very powerful, but one of the main challenges is to maintain the feasibility of solutions. Evolutionary approaches for nurse scheduling problems usually deploy some kind of local search as mutation and/or repair methods to generate feasible solutions or even improve the quality of solutions.

Even though, there were many models and approaches proposed for the nurse scheduling problems in the literature, there is still a big gap between nurse scheduling in research and the challenging requirements in real hospital environment [24]. The reason behind this is that models in research are often a much simplification of real-world nurse scheduling problems. The current trend is to address the requirement of the real world [24] and try to bridge the gap between research models and real-world models which could be pursed by including as many constraints in the research models but still allowing flexibility of models.

It is noted that it is difficult to compare different approaches for nurse scheduling problems due to a wide range of nurse scheduling models deployed by researchers. Each approach is normally designed to solve only a few or even one specific model. There is a trend of deploying the hybridisation of different approaches to solve nurse scheduling problems. Burke et al. [24] pointed out that interactive approaches, which incorporate problem specific methods and heuristics, are the current state-of-the-art in solving complex real-world nurse scheduling problems.

Chapter 3

Evolutionary Multiobjective Optimisation

Many real-world optimisation problems are multiobjective and the application of heuristic and evolutionary techniques to solve these kind of problems is a very active research area and has increased considerably over the last decade or so. The main challenge in solving multiobjective optimisation problems is that several criteria (usually conflicting) need to be taken into consideration simultaneously to assess the quality of a solution. Confronted with a multiobjective optimisation problem, a decision-maker should find an appropriate solution that represents a good compromise between the several possibly conflicting objectives. As aforementioned in chapter 1, there are three approaches to find such a solution, a-priori, interactive and a-posteriori preference articulations. The a-priori approach requests the preference amongst objectives before the search. However this is a non-trivial task in real-world optimisation problems. The techniques deployed by the a-priori approach include lexicographic ordering, linear aggregating function and nonlinear aggregating function [33]. The lexicographic ordering technique ranks objectives in order of importance then the optimum solution is obtained by optimising objective functions in a sequential manner, starting with the most important one.

The linear aggregating function technique uses weighting coefficients to compute solution's fitness which is then optimised. This coefficient vector represent the relative importance of objectives. The nonlinear aggregating function technique defines a suitable probability of acceptance (or goal) to be achieved for objectives. The main advantage of the a-priori approach is its simplicity and computational efficiency. However, this approach tends to prefer certain objectives in the problem. It could lead the population to converge to a particular part in the search space. This approach is most suitable when the importance (or the goal) of each objective is clearly defined. However, if the goal of the objectives is not clearly defined by the decision makers, the interactive approach could be appropriate. The interactive preference articulation could be referred to as the process of finding an intermediate solution then obtaining response from the decision maker regarding this solution to alter the preferences of the objectives accordingly. This process is repeated until the decision maker is satisfied or no further improvement is possible [33]. The interactive approach could be also ideal if only a certain region of the search domain is of interest. The last approach, the a-posteriori preference articulation, does not require any preference over the objectives. Regarding this approach, it is not easy to select just one solution because there are usually more than one good tradeoff solutions. Instead of find a single solution, it is required to generate a set of solutions known as a "front" or "trade-off" [32]. In Pareto based multiobjective optimisation, a set of non-dominated solutions, also known as a Pareto set, is sought so that the decision-maker can select the most appropriate solution. Using Pareto dominance, a solution x is said to be non-dominated if there is no other solution that is better than x for at least one objective and as good as x in the other objectives.

Evolutionary algorithms seem especially suitable to deal with Pareto based multiobjective optimisation problems because they can produce multiple promising solutions in a single run. The term evolutionary algorithm (EA) is derived from Darwin's concept of natural evolution in biological organisms [37]. Every individual

in the population has a chance of surviving and reproducing its genetic material into future generations. Following the analogy to biological evolution, an EA evolves a population of solutions or individuals to find a good set of solutions to the optimisation problem at hand. The evolution in an EA is based on the recombination and mutation of selected parents to produce offspring. High quality offspring could replace parents and compete for survival and reproduction in next generations.

This chapter provides the basic concepts of multiobjective optimisation problems (MOPs) and evolutionary multiobjective optimisation (EMO). It reviews some of important and recent evolutionary multiobjective algorithms. Key issues in EMO algorithms are also discussed. This chapter suggests some direction in designing a good evolutionary multiobjective algorithm which should be able to obtain Pareto fronts that are both well-distributed and well-converged. Finally, a set of performance metrics to assess EMO algorithms is presented.

3.1 Multiobjective Optimisation

The MOP requires to find a n-decision variable vector $\vec{x}$ that optimises the m-objective vector $f(\vec{x})$. Without the loss of generality, let us assume that the MOP is a maximisation problem.

Definition 1. (Multiobjective Optimisation Problem): *given the decision vector $\vec{x} = (x_1, x_2, \ldots, x_n)^T$ which belongs to the feasible region S formed by constraint conditions, maximise the objective vector $\vec{x}$:*

$$\begin{aligned} \text{maximise} \quad & \vec{f}(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), \ldots, f_m(\vec{x})) \\ \text{subject to} \quad & \vec{x} \in S \end{aligned} \tag{3.1}$$

In order to find solutions to the MOP, the optimiser requires mechanisms to compare solutions to identify superior solutions. One of the classical methods is the weighted-sum approach which scalarises the m-objectives into a single objective represented by a scalar value:

$$F(\vec{x}) = \sum_{i=1}^{m} w_i f_i(\vec{x}) \tag{3.2}$$

where w_i is the weight of the ith objective function. It is common that $\vec{w}$ is chosen as a normalised weight vector ($\sum_{i=1}^{m} w_i = 1$). This is the simplest and probably the most widely accepted method [45]. However, it is controversial to choose an appropriate normalised weight vector $\vec{w}$ because the m-objectives could be non-commensurable. Additionally, choosing $\vec{w}$ is equivalent to defining a priori preferences over the m-objectives before the search which is also not an easy task.

Recent multiobjective optimisation algorithms deploy a different approach, the use of a domination concept, which was proposed by Vilfredo Pareto in 1896 [101]. In the last two decades or so, Pareto dominance has been widely adopted as the technique to establish the superiority between solutions in multiobjective optimisation.

In Pareto dominance, a solution $\vec{x}$ is considered to be better than a solution $\vec{x^*}$ if and only if the objective vector of $\vec{x}$ dominates the objective vector of $\vec{x^*}$:

Definition 2. (Pareto Dominance): *A solution $\vec{x}$ in the search space S dominates a solution $\vec{x^*}$ in the search space S ($\vec{x} \succeq \vec{x^*}$) if and only if $\vec{x}$ is not worse than $\vec{x^*}$ in all objectives ($f_i(\vec{x}) \geq f_i(\vec{x^*})$ $\forall i = 1, \ldots, m$) and $\vec{x}$ is strictly better than $\vec{x^*}$ in at least one objective $i = 1, \ldots, m$ ($f_i(\vec{x}) > f_i(\vec{x^*})$).*

In the context of Pareto dominance, we can further distinguish between *weak dominance* and *strong dominance* [27] or *loose dominance* and *strict dominance* [38] respectively.

Weak dominance. Pareto dominance is sometimes simply referred to as a weak dominance. A solution $\vec{x}$ weakly dominates a solution $\vec{x^*}$ ($\vec{x} \succeq \vec{x^*}$) if $\vec{x}$ is better than $\vec{x^*}$ in at least one objective and is as good as $\vec{x^*}$ in all other objectives.

Strong dominance. A solution $\vec{x}$ strongly dominates a solution $\vec{x^*}$ ($\vec{x} \succ \vec{x^*}$) if $\vec{x}$ is strictly better than $\vec{x^*}$ in all objectives.

Non-dominance. If neither $\vec{x}$ dominates $\vec{x^*}$ nor $\vec{x^*}$ dominates $\vec{x}$, then both solutions are said to be incomparable or non-dominated. In this case, no solution is clearly preferred over the other, at least under the Pareto dominance criterion.

The set $\mathcal{P}^*$ consisting of all non-dominated solutions for a given set of solutions P is called the non-dominated set of P. Solutions in $\mathcal{P}^*$ are not dominated by any solutions in P. The non-dominated set $\mathcal{P}^*$ is also termed as the *Pareto set* of P which refers to the decision variable space. In the literature, there is also another term known as the *Pareto front* of P which refers to the objective functions space.

Definition 3. (Pareto Set): *For a given MOP and its set of solutions P, the Pareto set $\mathcal{P}^*$ is defined as follows:*

$$\mathcal{P}^* = \left\{\vec{x} \in P | \neg\exists \vec{x^*} \in P : \vec{x^*} \succeq \vec{x}\right\} \tag{3.3}$$

When the set P is the entire search space ($P = S$), the *Pareto set* $\mathcal{P}^*$ of the set P is called the *Pareto-optimal set.*

Definition 4. (Pareto-Optimal Set): *For a given MOP and its search space S, the Pareto-optimal set $\mathcal{PF}$ is defined as follows:*

$$\mathcal{PF} = \left\{\vec{x} \in S | \neg\exists \vec{x^*} \in S : \vec{x^*} \succeq \vec{x}\right\} \tag{3.4}$$

The procedure to find the Pareto set $\mathcal{P}^*$ for the set of solutions P is as follows [45]:

Step 1 Initialise $\mathcal{P}^* = \{\vec{x_1}\}$. Set counter $i = 2$ for solutions in P.

Step 2 Set $\vec{x_j}$ is the first non-dominated solution in $\mathcal{P}^*$.

Step 3 Perform domination check for solution $\vec{x_i}$ in P and non-dominated solution $\vec{x_j}$ in $\mathcal{P}^*$.

Step 4 If $\vec{x_i}$ dominates $\vec{x_j}$, delete non-dominated solution $\vec{x_j}$ from $\mathcal{P}^*$, set $\vec{x_j}$ as the next non-dominated solution in $\mathcal{P}^*$ then go to Step 3. If $\vec{x_j}$ dominates $\vec{x_i}$, increase i by one then go to Step 2. If $\vec{x_i}$ and $\vec{x_j}$ are non-dominated, set $\vec{x_j}$ as the next non-dominated solution in $\mathcal{P}^*$ then go to Step 3. For all three cases, if the last non-dominated solution in $\mathcal{P}^*$ is reached, hence there is no more $\vec{x_j}$ solution to perform Step 3, then go to Step 5.

Step 5 Insert $\vec{x_i}$ into $\mathcal{P}^*$ ($\mathcal{P}^* = \mathcal{P}^* \bigcup \{\vec{x_i}\}$). If $i < |P|$, increase i by one then go to Step 2. Otherwise terminate and declare $\mathcal{P}^*$ as the Pareto set.

Although the conventional Pareto dominance has been extensively used in the literature as the criterion to compare the fitness of solutions in multiobjective optimisation, there are proposals for alternative forms of Pareto dominance, known as *relaxed Pareto dominance* [84, 86] also called 'extended' dominance. These variations of Pareto dominance aim to find solutions in difficult areas (like the extremes of the tradeoff surface) or attempt to combine convergence and diversity in order to improve the performance of multiobjective optimisers and achieve a better Pareto set in difficult problems. Usually in these relaxed or extended dominance relationships, some transferring functions are applied to the m-objective values of the objective vector $\vec{f}(\vec{x})$ before comparing the solutions. In general, relaxed Pareto dominance allows a solution $\vec{x}$ to dominate another solution $\vec{x^*}$ for which $\vec{x}$ and $\vec{x^*}$ are Pareto non-dominated solutions or even $\vec{x}$ is Pareto-dominated by $\vec{x^*}$. It has been shown that *relaxed Pareto dominance* helps to obtain better quality Pareto fronts in some problems (e.g. [82, 79, 27]). More details on *relaxed Pareto dominance* will be discussed in chapter 5.

3.2 Evolutionary Algorithms

The evolutionary algorithm stands for a class of stochastic optimisation technique inspired by the principles of natural selection and natural genetics [52, 106]. EMO is the technique using EAs to solve MOPs [122, 44, 33]. Colleo Coello [33] pointed out that the potential of EAs for solving MOPs was suggested by Rosenberg in his PhD thesis [105]. Since then, there has been an enormous amount of work dedicated to this field. The first acknowledged EMO algorithm, by Schaffer [107], is the *Vector Evaluation Genetic Algorithm* (VEGA) which attempted to solve machine learning problems. During the 1990s, there were several EMO algorithms proposed in the field such as MOGA by Foncesca and Flemming [53], NPGA by Horn et al. [61], NSGA by Srinivas and Deb [110], SPEA by Zitzler and Thiele [125]. In the early 2000s, there was a surge of publications describing EMO algorithms including, but not limited to, PAES by Knowles and Corne [77], PESA2 by Corne et al. [36], NSGA2 by Deb et al. [47], SPEA2 by Zitzler et al. [124], MOGLS by Jaszkiewicz [71], ϵ-MOEA by Deb et al. [46], SEAMO2 by Mumford [96] and IBEA by Zitzler and Künzli [123]. The contribution to the EMO field continues to grow with recent publications such as MOEA/D by Zhang and Li [121], and other hypervolume-based approaches such as SMS-EMOA by Beume et al. [12], SIBEA by Brochhoff and Zitzler [16] among many others.

According to Deb [41], the EMO field was judged as one of the three fastest growing field of research and application among all computational intelligence topics in recent years. Despite the fact that EMO research and application has grown considerably [41], there is still strong interest in further improving the performance of EMO and exploring new challenging methodologies to attempt to find optimal solutions to multiobjective problems [91]. Perhaps the most noticeable research direction nowadays is to enhance EMO methods by incorporating concepts from different fields which results in sub-fields of research within the EMO framework. Examples of these sub-field are memetic algorithms and preference-based EMO

algorithms. Some important and recent EMO algorithms in the literature are reviewed below. Other earlier EMO algorithms could be found in several review papers [31, 32, 34].

3.2.1 Overview of Evolutionary Algorithms

Within the EA framework, individuals (solutions to the optimisation problem) are encoded as *chromosomes*, a collection of genes, which represents the characteristics of that individual. The *chromosome* is also known as the *genenotype*. The *phenotype*, which represents the quality of the individual, is defined by decoding the *chromosome*. The evolutionary process starts by evaluating the quality of individuals in the population. Individuals in the current population, known as parents, then go through selection for the recombination process. Selected parents are recombined to create offspring. The replacement process then chooses high quality individuals (i.e. parents and offspring) for the population of the next generation. These steps repeat until the stopping criteria is met. The conceptual evolutionary algorithm is as follows:

ConConceptual Evolutionary Algorithm

Initialisation: create an initial population

Evaluation: evaluate fitness values of individuals in the population.

while stopping criteria not met **do**

Selection: choose high fitness value parents to fill the mating pool.

Variation: apply variation operators on parents to create offspring.

Evaluation: evaluate fitness values of parents and offspring.

Replacement: replace parents with higher quality offspring.

end while

output the best individuals in the population

3.2.1.1 Individual Representation

Within the EA framework, individuals are considered at two levels: the *genotypic* level and the *phenotypic* level [52]. The *genotype* of an individual, also referred to as the *chromosome*, is a string of *genes* of finite length. The chromosomes could be in any form such as binary, integer/alphabet, real-valued. The most common forms are binary (binary representation) and integer/alphabet (e.g. permutation representation). In the binary representation, each gene in encoded individuals takes a value of either 0 or 1 and the value of genes is important. In the permutation representation, each gene in encoded individuals take a distinguished value and the position of genes is important. The choice of chromosome representation, which depends on the problem as well as the EA itself, is important. An unsuitable representation could lead to unnecessary computational overhead and low performance of the EA.

The genenotype represents the characteristics (decision variables) of individuals but not the quality of individuals. The quality of individuals, the phenotype, is obtained by decoding the chromosome. The phenotype is also known as the objective value of individuals. It is crucial to understand the differences between genotype and phenotype representations because of their different purposes and association to different stages during the evolutionary process.

3.2.1.2 Evaluation of Individual Fitness

The evaluation of an individual is the process which obtains the objective values, the phenotype, of the individual by decoding its genotype. The phenotype of individuals could be used to directly compare individuals based on Pareto dominance and at different stages of the algorithm such as in SEAMO/SEAMO2 [115, 96]. However, it is very common that the algorithm takes one step further in assessing individuals' quality by assigning a fitness to each individual. This process is re-

ferred to as the individual's fitness evaluation to distinguish from the individual's objective values evaluation. The fitness evaluation derives the fitness of an individual from its objective values. While computing the fitness of an individual, the fitness evaluation might or might not consider other individuals in the population. The fitness evaluation strategy is one of the core features of which could differentiate one EA from others. Fitness assignment strategies could be categorised into three types:

Dominance-based: The fitness of individuals is determined by comparing individuals to others in the population based on Pareto dominance (or other types of relaxed Pareto dominance); for example, NSGA2 and SPEA2.

Non dominance-based: The fitness of individuals is determined by applying transferring functions on the objective values of individuals which combine and/or modify the objective values; for example, MOEA/D.

Hybridisation: Based on some satisfaction on dominance condition, transferring functions are applied to the objective values of individuals to obtain the fitness value. For example, IBEA.

3.2.1.3 Parents Selection

The parents selection, or mating selection, is the process of selecting individuals to participate in the production of offspring [52]. There are two common mating selections: stochastic selection and tournament selection. Stochastic selection select parents at random regardless of parents' quality/fitness. Tournament selection applies an additional layer onto the stochastic selection. In tournament selection, parents are also drawn at random but selected parents then undergo a fitness comparison process. The highest quality parent, i.e. the winner of the tournament, is selected to be in the mating pool. The size of the tournament is usually set to 2 (binary tournament). There are other schemes such as fitness proportionate selec-

tion and truncation selection. In fitness proportionate selection, the probability of each individual being selected for the mating pool is in proportion to its fitness. In truncation selection, each of the top $(1/c)\%$ individuals in the population (with respect to fitness) get c copies in the mating pool. The selection of parents is also one of the key feature in EAs.

3.2.1.4 Replacement Strategy

The replacement strategy, which is also known as the environmental selection or survival selection, is the process of selecting individuals for the next generation based on the fitness of individuals. As opposed to the parents selection, which is usually stochastic, the replacement strategy, which usually select the best individuals based on their fitness, is mainly deterministic [58]. The replacement strategy is categorised into *generational* and *steady-state* selection schemes. The difference between these two schemes is at which point parents and offspring compete for survival. In the *generational* selection scheme, for each generation, the offspring population is constructed (based on the parent population) first. The sizes of the offspring population and parent population are usually the same size of the current population. The offspring population and the current population are combined. Individuals from resulting population are then selected for the next generation. However, the *steady-state* selection scheme allows offspring to compete for survivor and reproduction as soon as they are created. In other words, offspring are considered to replace their parents (or other individuals) in the current population immediately after offspring are constructed. Besides the fitness assignment and the parents selection, the replacement strategy is also one of the important aspects of an EA.

3.2.1.5 Reproduction Mechanism

The reproduction mechanism uses the genetic material, the genotype, of parents to create offspring. The aim of the reproduction mechanism is to manipulate the gene structure of individuals in the current population, create new individuals with the expectation that better individuals could be obtained. There are a large number of genetic operators that can be used for the reproduction, but broadly these are divided into two categories:

Recombination: is usually a binary operator which combines the genetic material from two parents to produce the genetic material for one or two offspring. The recombination operator, which is usually a random operator, decides which part of the genetic material from parents is inherited by the offspring. The principle behind the recombination is that "good" genes in parents are combined in the genetic material of the offspring.

Mutation: is a unary operator which is applied to the genotype of one individual in order to slightly modify the gene structure of that solution. The mutation operator is most times a stochastic operator. The mutation operator attempts to introduce a few new features that might not be inherited from the parents. The purpose is to add diversity to the population and contribute so that the entire search space can be possibly explored [106].

It is noted that both recombination and mutation operators are genotypic representation dependent. Different individual representations often require different operators. It is also noted that while operators for mutation are usually problem independent, many recombination operators are problem specific. A few problem independent operators are presented below:

The simplest form of mutation operator is *bit-flip* mutation used with a binary representation. Each gene in the genetic material of an individual is inverted

(between 0 and 1) with certain probability (usually very low). This probability is known as the mutation probability, which is relatively small to prevent too much disruption of the inherited genetic material. For a permutation representation, the mutation operator usually swaps two arbitrarily selected genes in the permutation list at once or more times.

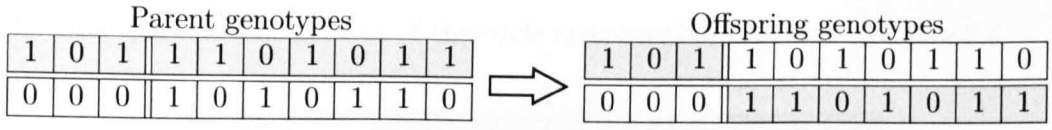


Figure 3.1: One Point Crossover

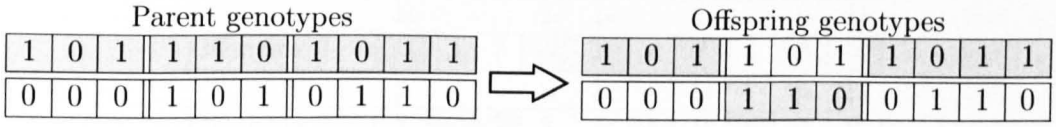


Figure 3.2: Two Point Crossover

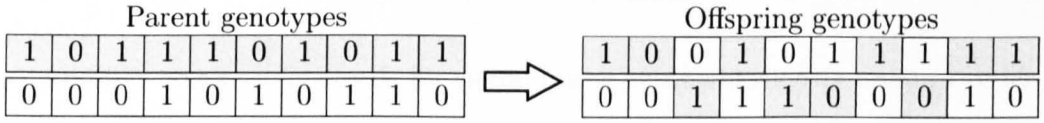


Figure 3.3: Uniform Crossover

Recombination, or crossover, operators have many different forms. Two common forms of crossover for binary representation are *k-point* and *uniform* crossover. The simplest versions of *k-point* crossover are one-point crossover (figure 3.1) and two-point crossover (figure 3.2). In one-point crossover, a crossover point is selected at random then genes on one side of the crossover point are exchanged between individuals. In two-point crossover, two crossover points are selected at random and genes between two crossover points are exchanged between individuals [106]. In uniform crossover (figure 3.3), every gene is exchanged between individuals with certain probability, which is usually set to 0.5.

With respect to the permutation representation, crossover operators must preserve the genetic information i.e. all genes must be in the genotype in which ever way the permutation list is modified. Oliver et al. [98] presented cycle crossover, applied on two parents, which preserves the absolute position of genes in the permutation list. A crossover point is selected at random as the starting position of

the cycle. The gene at that position of the selected parent is copied to the genotype of the offspring (5). The next position of the cycle is the position, in the selected parent, of the gene which appears at the current position in the other parent (8). The cycle repeats until the gene at the starting position of the cycle is encountered. Unfilled positions of the offspring are filled by genes, in those positions, of the other parent. The process of the cycle crossover is illustrated in figure 3.4.

				⇓						
Parent 1	0	7	1	5	6	3	8	4	9	2
Parent 2	2	3	7	8	5	4	9	1	0	6
	$\downarrow_4$			$\downarrow_1$	$\downarrow_6$		$\downarrow_2$		$\downarrow_3$	$\downarrow_5$
Offspring 1	0	3	7	5	6	4	8	1	9	2
	$\downarrow_4$			$\downarrow_1$	$\downarrow_2$		$\downarrow_6$		$\downarrow_5$	$\downarrow_3$
Offspring 2	2	7	1	8	5	3	9	4	0	6

Figure 3.4: Cycle Crossover

Another crossover operator for permutation representation is uniform order-based crossover, applied on two parents. Uniform order-based crossover requires a random binary template. Firstly, the offspring is filled with genes from a selected parent at positions indicated as 1 in the template. The remaining genes in the selected parent, which are sorted in the same order as appeared in the other parent, then fill vacant positions in the offspring. The uniform order-based crossover is illustrated in figure 3.5.

Parent 1	0	7	1	5	6	3	8	4	9	2
Parent 2	2	3	7	8	5	4	9	1	0	6
Template	1	0	0	1	1	1	0	1	0	0
Offspring 1	0	2	7	5	6	3	8	4	9	1
Offspring 2	2	0	7	8	5	4	6	1	3	9

Figure 3.5: Uniform Order-Based Crossover

In the literature, there are also other popular crossover operators for permutation representation such as partially mapped crossover [57], position based crossover [112], order crossovers [39, 112]. These crossover operators and their comparison could be found in [111]. Throughout this thesis, one-point crossover

operator is employed for binary representation and cycle crossover is for permutation representation.

3.2.1.6 Stopping Criteria

Stopping criteria define states in which the evolutionary search terminates and the best individuals are presented. Stopping criteria usually vary accordingly to the type of applications and the purpose of the studies. For theoretical studies, in which the purpose is to investigate the performance of newly proposed EAs for example, the stopping criteria are usually the number of evaluations (or generations) or the amount of execution time. However, the latter is rarely used due to its low reliability and dependence on other factors such as hardware, operating systems and programming languages. For real-world application (especially in real-time application), in which the computational time is limited, it is sensible to set the amount of execution time as the stopping criterion. There are other criteria such as on-line performance metrics which keep track of the improvement of population or best solutions until no improvement after a certain amount of time or evaluations. The number of generations could be also the stopping criterion of real-world applications.

3.2.1.7 Other Issues in EA

The core features of an EA have been discussed above. Although, when designing an EA, it is also worthwhile to pay attention to other issues in order to design a good performing EA. These issues include, but are not limited to:

Population Initialisation & Reinitialisation. The simplest approach is to generate random individuals for the initial population. However, it is possible and often advisable to use heuristics to construct (better) individuals. The heuristics are usually knowledge-based and problem dependent. Apart from being generated

at the start of the search, new individuals could also be introduced during the search by heuristics instead of using reproduction operators as discussed above. This is known as reinitialisation in which a part or whole population is reinitialised if the search stagnates.

Generational vs. Steady-State. As aforementioned, the replacement strategy could be either a generational or a steady-state approach, also referred to as generational or steady-state selection. The decision on which approach is deployed within an EA is mainly driven by the fitness assignment strategy. If the fitness assignment strategy requires all (or a large number of) individuals in the population to estimate the fitness of an individual, the generational approach should be employed. On the other hand, the steady-state selection could be used if the fitness of an individual is not effected by other individuals in the population. Inappropriate approaches could lead to expensive computational time and degraded performance of the search. It is noted that recent studies show a better performance of the steady-state selection over generational selection. For example, Durillo et al. [49] modified NSGA2 and SPEA2 from generational to steady-state selection and reported better performance than the original algorithms although the computational time increased significantly. Recent steady-state selection EAs, such as SEAMO2 [96], SMS-EMOA [12], MOEA/D [121] also report better performance than generational state-of-the-art EMO algorithms such as NSGA2 and SPEA2. Therefore EMO algorithms designers should pay attention to this issue.

Exploration vs. Exploitation. This is also known as diversity vs. intensification of the population. The reproduction mechanism is mainly the driving force for exploration whereas exploitation is taken care by the replacement strategy. In EAs, it is very difficult to obtain good results in terms of both exploration and exploitation at the same time. There is usually a trade-off between these two aspects. Therefore, balancing well this trade-off could lead to high performance EMO algorithms.

Elitism. This term means that the best individuals are always included in the next population. It is unambiguous in the single-objective framework where there is only one best individual at anytime. However under the multiobjective framework, there could be always more than one best individuals. The number of best individuals could increase dramatically with respect to the number of objectives of the problem. With a limited population size, it is non-trivial to identify which best individuals should be kept. An external archive or favouring best solutions with at least one best objective value could be the answer. Laumanns et al. [83] also argued that the usefulness of elitism strongly depends on the mutation rate. However, in the author's opinion, it remains inconclusive how elitism should be tackled.

Duplication. Duplication occurs in the population if there is at least a pair of distinct (in the decision space) individuals having the same objective values. The problem with allowing duplication is that all individuals in the population could converge too quickly to a single point in the objective space which is not desirable. To deal with this problem, a hard approach it to strictly not allow any duplication (like in SEAMO2) while in a soft approach is to allow the replacement strategy to eliminate duplication (like in NSGA2 and SPEA2).

Mating Restriction. The idea of restricted mating comes from natural selection where mating only seems to happen between similar individuals. Deb and Goldberg [42] argued that mating between dissimilar individuals will likely lead to unfit offspring called *lethals*. However, mating between too similar individuals, known as *in-breeding*, could affect adversely the progress of the search. Mating restriction could be performed on either the genotype or phenotype of individuals. In the literature, mating restriction is usually controlled by a mating radius σ_{mating}. There are a few studies which support the case of mating restriction on similar individuals [40, 66, 67, 85]. Chapter 4 will present deeper discussion on mating restriction.

3.2.2 Multiobjective EAs in the Literature

3.2.2.1 Non-dominated Sorting Genetic Algorithm (NSGA)

Srinivas and Deb [110] proposed NSGA which ranked the population into fronts based on Pareto non-domination and each front is assigned an increased dummy fitness value. These classified individuals share dummy fitness values to maintain the diversity of the population. This process continues until all individuals in the population are classified. Later, Deb et at. [47] proposed a fast non-dominated sorting approach to classify individuals in a population into different non-domination levels or different fronts (NSGA2). NSGA2 employed a density estimation metric to preserve the population diversity. It required to sort each front according to each objective function value in ascending order of magnitude. The boundary individuals are assigned an infinite distance value. The density of individuals surrounding other intermediate individuals in the front is the sum of the average normalised distance of two individuals on either side of this individual along each of the objectives. The best fronts of the offspring population and the archive combined are selected for the next archive. If the size of the archive is greater than N, individuals in the last front are removed to reduce the size of the archive to N based on their crowding value. NSGA2 uses a fixed population and a fixed archive of size N. The mating pool is filled by binary tournament selection with the following priority: non-domination level then crowding distance. The offspring population is formed by applying crossover and mutation on the mating pool. There is no treatment to prevent duplication of individuals whilst forming the archive. Duplicated individuals could be spotted by the crowding value and removed during the truncation of the archive but this process is not exhaustive and hence duplicated solutions may remain.

3.2.2.2 Strength Pareto Evolutionary Algorithm (SPEA)

Zitzler and Thiele [125] proposed SPEA which maintains a population $\boldsymbol{P}$ and stores all non-dominated individuals in an external archive $\overline{\boldsymbol{P}}$. Each individual in $\overline{\boldsymbol{P}}$ is assigned a strength value proportional to the number of individual in $\boldsymbol{P}$ that it dominates. The strength value of $\vec{x} \in \overline{\boldsymbol{P}}$ is:

$$s(\vec{x}) = \frac{\left|\{\vec{x^*} \in \boldsymbol{P} : \vec{x} \succ \vec{x^*}\}\right|}{|\boldsymbol{P}| + 1} \tag{3.5}$$

and the fitness of $\vec{x^*} \in \boldsymbol{P}$ is the sum of the strength value of all individuals in $\overline{\boldsymbol{P}}$ that dominate $\vec{x^*}$:

$$f(\vec{x^*}) = 1 + \sum_{\vec{x} \in \overline{\boldsymbol{P}} : \vec{x} \succ \vec{x^*}} s(\vec{x}) \tag{3.6}$$

SPEA deploys the average linkage clustering approach to reduce the size of the external archive $\overline{\boldsymbol{P}}$, if necessary. SPEA allows individuals from both the population $\boldsymbol{P}$ and the external archive $\overline{\boldsymbol{P}}$ to be selected for reproduction.

Later, Zitzler et at. [124] improved SPEA by employing a fine-grained fitness assignment strategy which for each individual takes into account how many individuals it dominates and it is dominated by. Each individual in both the archive and the population is assigned a strength value $S(\vec{x})$, indicating the number of individuals it dominates. The raw fitness $R(\vec{x})$ of an individual is defined as the sum of the strength value of all individuals by which it is dominated.

$$S(\vec{x}) = \left|\left\{\vec{x^*} \in \boldsymbol{P} \bigcup \overline{\boldsymbol{P}} : \vec{x} \succ \vec{x^*}\right\}\right| \tag{3.7}$$

$$R(\vec{x}) = \sum_{\vec{x^*} \in \boldsymbol{P} \bigcup \overline{\boldsymbol{P}} : \vec{x^*} \succ \vec{x}} S(\vec{x^*}) \tag{3.8}$$

SPEA2 uses an adaptation of the k-th nearest neighbour method to estimate the density of an individual. The density of an individual is estimated as the inverse of the distance to the k-th nearest neighbour. Then, the density is added to the

raw fitness $R(\vec{x})$ to yield the fitness of individual $\vec{x}$. SPEA2 fills the mating pool using binary tournament selection only on the archive. Crossover and mutation is applied on the mating pool to create the offspring population. All individuals in the offspring and the archive population having fitness less than one, i.e. all non-dominated individuals in $\boldsymbol{P} \bigcup \overline{\boldsymbol{P}}$, are copied to the next archive. Zitzler et at. pointed out that there are two situations: either the archive is too small or too large. If the archive is too small, the best dominated individuals, based on the strength, are copied to fill $\overline{\boldsymbol{P}}$. In the later situation, non-dominated individuals in the archive are iteratively removed until the archive size is not exceeded. The removal of non-dominated individuals from the archive is carefully managed by using an archive truncation method that guarantees the preservation of boundary solutions. As in NSGA2, SPEA2 relies on the archive truncation to remove duplicated individuals but it does not guarantee that the archive contains no duplication.

3.2.2.3 Simple Evolutionary Algorithm for Multi-objective Optimization (SEAMO)

Mumford (Valenzuela) proposed SEAMO [115] as a steady-state population and a simple elitist replacement strategy. Each individual in the population acts as the first parent once and a second parent is chosen at random. Offspring is produced by applying a crossover on the pair of parents, followed by a single random mutation. The offspring replaces one of the parents if either its objective vector improves on any *best-so-far* objective function or it dominates that parent. SEAMO does not allow duplication in its population. Therefore, any duplicated offspring dies before the replacement process. Mumford further investigated other replacement strategies and proposed SEAMO2 [96]. The difference between the replacement strategy in SEAMO2 and SEAMO is that in SEAMO2, if neither the offspring dominates the parents nor the parents dominate the offspring, the offspring replaces a random individual in the population that the offspring dominates. SEAMO2 and

SEAMO do not use any individual fitness assignment or niching technique. These algorithms heavily rely on the replacement strategy to maintain diversity (i.e. the replacement of the *best-so-far* solutions with offspring that improve on *best-so-far*) and pursuit convergence (i.e. the replacement of dominated parents with offspring).

3.2.2.4 Indicator Based Evolutionary Algorithm (IBEA)

Zitzler and Künzli [123] proposed a general framework indicator-based evolutionary algorithm. IBEA could be referred to as an EMO algorithm but guided by general preference information of the dominance relationship, a *binary quality indicator* $I\left(\{\vec{x^*}\},\{\vec{x}\}\right)$. IBEA uses a fixed size archive and an offspring population. Offspring is produced by recombination and mutation on a pair of parents selected from the archive. The archive and the offspring population is combined and truncated to generate the new archive for the next iteration. IBEA needs neither specifications of weights or targets (as in aggregation methods) nor the dominance relationship and distribution techniques (as in other EMO algorithms). IBEA uses this *binary indicator* guiding the search to generate the approximation set. The *binary quality indicator* in IBEA maps an ordered pair of individuals to a real number which therefore could be used for fitness calculation $F(\vec{x})$.

$$F(\vec{x}) = \sum_{\vec{x^*} \in P \setminus \{\vec{x}\}} e^{-I(\vec{x^*},\vec{x})/\kappa} \tag{3.9}$$

where κ is the fitness scaling factor. Zitzler and Künzli [123] proposed two indicators, the additive ϵ-indicator $I_{\epsilon+}$ and the hypervolume-indicator I_{HD}. For an ordered pair of individuals $(\vec{x^*},\vec{x})$, the $I_{\epsilon+}(\vec{x^*},\vec{x})$ indicator gives the minimum distance for which $\vec{x^*}$ is translated in each dimension in objective space to weakly dominate $\vec{x}$ while the $I_{HD}(\vec{x^*},\vec{x})$ indicator measures the hyperspace volume that is dominated by $\vec{x}$ but not by $\vec{x^*}$ with respect to a predefined reference point.

Zitzler and Künzli [123] also pointed out that the indicator values $I(\vec{x^*}, \vec{x})$ could be widely spread for different problems making it difficult to approximate the value for κ. Therefore, they suggested adaptively scaling the indicator values to the range $[-1, 1]$ for all solutions in the population. Furthermore, to eliminate the issue of estimating a good reference point for the I_{HD} indicator, Zitzler and Künzli suggested also scaling the objective values to the range $[0, 1]$. Therefore the fitness calculation in the adaptive IBEA is modified as follows:

$$F(\vec{x}) = \sum_{\vec{x^*} \in P \setminus \{\vec{x}\}} e^{-I(\vec{x^*}, \vec{x})/(c \cdot \kappa)} \tag{3.10}$$

where $c = \max_{\vec{x^*}, \vec{x} \in P} \left\{ \left| I(\vec{x^*}, \vec{x}) \right| \right\}$ and $I(\vec{x^*}, \vec{x})$ is calculated based on the scaled objective values f_i' which is determined as follows:

$$f_i'(\vec{x}) = \left(f_i(\vec{x}) - \underline{b_i} \right) / \left(\overline{b_i} - \underline{b_i} \right) \tag{3.11}$$

the lower bound is $\underline{b_i} = \min_{\vec{x} \in P} \{ f_i(\vec{x}) \}$, the upper bound is $\overline{b_i} = \max_{\vec{x} \in P} \{ f_i(\vec{x}) \}$ and $\kappa = 0.05$ for all problems and indicators. See [123] for more details on IBEA.

3.2.2.5 $\mathcal{S}$-Metric Selection EMOA (SMS-EMOA)

Beume et al. [12] proposed SMS-EMOA as a steady state population and selection scheme. SMS-EMOA's explicit goal is to maximise the hypervolume, $\mathcal{S}$-metric value [125], of the population. At each iteration, an offspring is produced by recombination and mutation. The offspring replaces an individual in the population $\boldsymbol{P}_t$ if it leads to higher quality of the population with respect to the $\mathcal{S}$-metric by removing an individual which exclusively contributes the least hyper volume. The resulting population $\boldsymbol{P}_{t+1}$, formed by combining the offspring and the previous population $\boldsymbol{P}_t$, is partitioned into different non-dominated fronts using the fast non-dominated sorting algorithm NSGA2 [47]. The first front $\mathcal{R}_1$ contains all non-dominated solutions of $\boldsymbol{P}_{t+1}$, the second front contains all individuals that are

non-dominated in the set $(\boldsymbol{P}_{t+1} \backslash \mathcal{R}_1)$, etc. Finally, an individual r is discarded from the last front $\mathcal{R}_v$ (the worst ranked front) if that individual is the one contributing the least hyper-volume to this last front $\mathcal{R}_v$.

$$r = \arg\min_{s \in \mathcal{R}_v} \left[\Delta_{\mathcal{S}}\left(s, \mathcal{R}_v\right)\right] \tag{3.12}$$

$$\Delta_{\mathcal{S}}\left(s, \mathcal{R}_v\right) = \mathcal{S}\left(\mathcal{R}_v\right) - \mathcal{S}\left(\mathcal{R}_v \backslash \{s\}\right) \tag{3.13}$$

Beume et al. also investigated several variants of SMS-EMOA aiming to promote solutions to partially filled regions of the Pareto front and to reduce expensive computation of the hypervolume measure. They suggested that if there is more than one front in P_{t+1} $(v > 1)$, the selection based on equation 3.13 should be replaced by equation 3.14:

$$r = \arg\max_{s \in \mathcal{R}_v} \left[d\left(s, \boldsymbol{P}_{t+1}\right)\right] \tag{3.14}$$

$$d\left(s, \boldsymbol{P}_t\right) = \left|\{y \in \boldsymbol{P}_t | y \succ s\}\right| \tag{3.15}$$

The above means that the individual with the highest number of dominating points should be discarded. The purpose of this replacement strategy is to promote dominating individuals to rise in rank to better fronts and fill vacant (less populated) areas in the current Pareto front [12].

3.2.2.6 Multiobjective Evolutionary Algorithm Based on Decomposition (MOEA/D)

MOEA/D, proposed by Zhang and Li [121], decomposes a multiobjective optimisation problem into a number of scalar optimisation subproblems and optimises them simultaneously. Each subproblem is solved by utilising information from its neighbouring subproblems. All non-dominated individuals found during the search are stored in an external archive. MOEA/D predefines a set of N evenly spread

weight vectors $\left\{\vec{\lambda^1}, \ldots, \vec{\lambda^N}\right\}$, where N is the number of subproblems, or the population size of MOEA/D. Each ith subproblem corresponds to a weight vector λ^i. Each vector is the size of m (the number of objectives in the problem). N is determined based on a controlling parameter H which is explained more clearly in page 51. The neighborhood of the ith subproblem consists of all subproblems with the weight vectors closest to λ^i which include the ith subproblem itself. Then, the population of MOEA/D consists of the best solution $\vec{x}^i$ found so far for each ith subproblem. At each iteration, an offspring in the ith subproblem is produced by recombination and mutation on parents which are the current solutions to subproblems in the neighbourhood of the ith subproblem. The offspring replaces current solutions of the ith subproblem and its neighbouring subproblems if the fitness value of the offspring is better than that of these current solutions. The fitness of a solution $\vec{x}$ is based on Tchebycheff approach:

$$g^{te}(\vec{x} \mid \vec{\lambda^j}, z) = \max_{1 \leq i \leq m} \left\{\lambda_i^j \, |f_i(\vec{x}) - z_i|\right\} \tag{3.16}$$

where z is the reference point $z_i = \max\{f_i(\vec{x}) \mid \vec{x} \in \boldsymbol{P}\}$. For more details on MOEA/D see [121].

3.2.2.7 Other Classes of EMO algorithms

Recently, there have been a number of approaches proposed not only to further improve the performance of existing EMO algorithms but also to make EMO algorithms more applicable and successful on the real-world applications. Some of these proposed approaches are briefly reviewed next.

EMO Algorithm Local Search is a class of stochastic heuristics (also known as memetic algorithms) which combine evolutionary algorithms with one or more phases of local search to improve the quality of solutions [58]. Within the classical EMO algorithm's framework, offspring are usually produced by crossover opera-

tors followed by random mutations. Therefore, it is very common that in EMO algorithm Local Search, the random mutation is replaced by one or more local search phases. Local search strategies, such as hill-climbing, simulated annealing, tabu search, usually use problem specific operators/heuristics. There have been proposals for general memetic algorithms such as by Knowles [78], Jaszkiewics [72], Bosman and de Jong [15] as well as for problem specific such as personnel scheduling, timetabling, job-shop scheduling [26, 18, 60]. See [59] for more details.

The ***epsilon*-MOEA** is based on the concept of ϵ-dominance proposed by Laumanns et al. [82]. The algorithm aims to improve the convergence and the diversity as well as the computational time [82, 46]. The underlying concept of this approach could be employed by any EMO algorithms by replacing the conventional Pareto dominance with ϵ-dominance. ϵ-dominance is discussed in section 5.2.

Preference-based EMO algorithm is a very recent research trend within the EMO algorithm framework. The main purpose of preference-based EMO algorithms is to deal with a large number of objectives. Problems with many objectives (usually more than 3) are of particular difficulty because when considering many objectives the size of the Pareto front increases substantially and it is difficult to generate and maintain all non-dominated individuals. This is because for many objectives, many solutions become mutually non-dominated. Another aim of preference-based EMO algorithms is to focus the search on particular parts of the Pareto front. A large number of objectives and the interest on particular parts in the search space are the two issues which are often encountered in real-world problems. The concept behind preference based EMO algorithms is to narrow the search space by incorporating preference information by the decision maker into the algorithm. There have been relatively few studies on this area, examples are those by Deb and Sundar [43], Deb et al. [48], Allmendinger et al. [5], Julian et al. [93], Thiele et al. [113], Wickramasinghe et al. [118]. The main approach to incorporate preference information in preference-based EMO algorithms is to define a reference

point in the objective space and restrict the search area around this point. At the time of writing, the most up-to-date contribution on preference-based EMO algorithms could that be of Julian et at. [93]. They proposed a variation of the Pareto dominance concept which facilitates the approximation of the efficient Pareto front around the desired area. Julian et at. also provided a simple strategy to update the reference point which allows the decision maker to interact with the search process.

3.3 Performance Assessment

The two main issues influencing the performance assessment of EMO algorithms are the chosen experimental problem and the set of performance metrics. The chosen problem should be understandable, easy to formulate, but yet difficult to solve [125]. Due to the multiobjective nature of the experimental problem, it require a set of performance metrics to assess the EMO algorithms' performance. This set of performance metrics should be able to address at least one of the following criteria for approximated Pareto sets obtained by MEOAs, which include, but are not limited to: the diversity, the convergence, the coverage and the distribution of solutions.

3.3.1 The 0/1 Multiple Knapsack Problem

Martello and Toth [88] formulated the single-objective knapsack problem as a set of n items and a knapsack. Each item has an associated weight (w_j) and an associated profit (p_j). The goal is to find a subset of n items that maximise the total profit. The total weight of this subset of n items must not exceed the knapsack's capacity (i.e. the subset of items fit into the knapsack). Martello and Toth [88] also defined a multiobjective knapsack problem which consists of n items, and m knapsacks.

Each knapsack has an associated capacity (c_i). The task is to find m disjoint subsets of items, each subset fits into a knapsack and maximise the total profit of selected items. This 0/1 multiple knapsack problem can be defined as follow:

$$
\begin{aligned}
\text{maximise} \quad & z = \sum_{i=1}^{m} \sum_{j=1}^{n} p_j x_{ij} \\
\text{subject to} \quad & \sum_{j=1}^{n} w_j x_{ij} \leq c_i, && \forall i = 1, \ldots, m \\
& \sum_{i=1}^{m} x_{ij} \leq 1, && \forall j = 1, \ldots, n \\
& x_{ij} = 0 \text{ or } 1
\end{aligned}
$$

There are also different proposals for the 0/1 multiple knapsack problem such as by Khuri et al. [75]. Different proposals reflect different real-world applications. However those proposals have not been widely adopted so far. It was not until the proposal by Zitzler and Thiele [125] for the 0/1 multiple knapsack problem. This formulation described by Zitzler and Thiele is the multiobjective version directly extended from the single-objective knapsack problem proposed by Martello and Toth [88]. Since then, this problem has been widely used as a benchmark problem for the performance assessment of not only EMO algorithms but also other search algorithms. The 0/1 multiple knapsack problem proposed by Zitzler and Thiele [125], which will be used in this thesis, is described below.

3.3.1.1 Problem Description

The multiple 0/1 knapsack is defined as follows: Given a set of n items, a set of m knapsacks, weight and profit associated with each item in a knapsack, and an upper bound for the each knapsack capacity. The goal is to find a subset of items that maximise the profit in each knapsack and the knapsack weight does not

exceed its capacity.

$$p_{i,j} = \text{profit of item } j \text{ in knapsack } i$$

$$w_{i,j} = \text{weight of item } j \text{ in knapsack } i$$

$$c_i = \text{capacity of knapsack } i$$

find a vector $\vec{x} = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$ such that:

$$\forall i \in \{1, 2, \ldots, m\} : \sum_{j=1}^{n} w_{i,j}.x_j \leq c_i \tag{3.17}$$

and maximise $f(\vec{x}) = (f_1(\vec{x}), f_1(\vec{x}), \ldots, f_m(\vec{x}))$, where

$$f_i(\vec{x}) = \sum_{j=1}^{n} p_{i,j}.x_j \tag{3.18}$$

and $x_j = 1$ if and only if item j is selected, otherwise $x_j = 0$.

3.3.1.2 Experimental Data

Zitzler and Thiele suggested 9 different sets of data including instances with two, three and four objectives in combination with 250, 500 and 750 items. Zitzler and Thiele also pointed out that the uncorrelated profits and weights, suggested by Martello and Toth [88], should be chosen when generating benchmark data for this problem. The profit and weight of an item j in knapsack i $(p_{i,j}, w_{i,j})$ are random integers in the interval $[10, 100]$. The capacity of a knapsack is half the total weight [125].

$$c_i = 0.5 \sum_{j=1}^{n} w_{i,j} \tag{3.19}$$

3.3.1.3 Problem Implementation

In the literature, there are three representations published for solutions to the 0/1 knapsack problem, binary representation, numeric representation and symbolic representation [95]. It is pointed out by Michalewicz and Arabas [90] that there are three possible capacity constraint handling techniques for the knapsack problem which are based on penalty functions, repair methods and decoders. These representation and constraint handling techniques could be directly extended to the 0/1 multiobjective knapsack problem. Reviews and comparisons on different representation and constraint handling techniques for the 0/1 multiobjective knapsack problem can be found in [35, 95]. This section only reviews the most two favoured combinations in the literature which are the order-based representation with a decoder and the binary representation with a heuristics repair methods.

Mumford (Valenzuela) deploys the order-based representation with a decoder for in the 0/1 multiple knapsack problem in SEAMO/SEAMO2 [115, 96]. The solution is represented as a simple permutation of items to be packed. The decoder starts from the beginning of the permutation list, packing an item one at a time until the weight for any knapsack exceeds its capacity. The packing stops and the last packed item is removed from all knapsacks. The offspring reproduction is based on the recombination on a pair of parents using cycle crossover [98] and followed by random mutation that consists of swapping two arbitrarily selected items within a single permutation list [115].

Another type of representation is the binary representation where solutions to the 0/1 multiple knapsack problem are represented as strings of 0s and 1s. Offspring are produced by applying one point crossover on a pair of parents followed by bit-flip mutation. Unlike the order-based representation in SEAMO/SEAMO2, where the decoder starts from the beginning of the permutation list, in a binary representation, it is highly likely that the total weight of selected items (indicated

as '1') exceeds the capacity of knapsacks and lead to infeasible solutions. Therefore, it requires a heuristic to repair infeasible solutions. Two heuristics to repair infeasible solutions for the 0/1 multiple knapsack problem represented as binary strings, are discussed below.

Zitzler and Thiele [125] proposed a greedy repair method that repeatedly removed items from the set of selected items until the capacity constraint is satisfied. Items are deleted based on the order determined by the *maximum profit/weight* ratio per item (q_j); for item j the *maximum profit/weight* ratio (q_j) is given by the equation (3.20)

$$q_j = \max_{i=1}^{m} \left\{ \frac{p_{i,j}}{w_{i,j}} \right\} \tag{3.20}$$

Items with lowest q_j are removed first until the capacity constraint is fulfilled. This mechanism diminishes the overall profit as little as possible [125].

Jaszkiewicz [72] used a weighted linear scalarising approach to repair infeasible solutions. Items are sorted according to the following ratio:

$$l_j = \frac{\sum_{i=1}^{m} \lambda_i \times p_{i,j}}{\sum_{i=1}^{m} w_{i,j}} \tag{3.21}$$

where $\vec{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_m)$ is the weight vector used in the current iteration. Later, Jaszkiewicz improved this greedy repair method which was subsequently deployed by Zhang and Li [121] as follows: Let set $J = \{j \mid 1 \leq j \leq n \wedge x_j = 1\}$ is the set of selected items and set $I = \left\{i \mid 1 \leq i \leq m \wedge \sum_{j=1}^{n} w_{i,j} \times x_j > c_i\right\}$ is a set of overfilled knapsacks. Repeatedly select item $k \in J$ to remove until none of the knapsack is overfilled, such that:

$$k = \arg\min_{j \in J} \frac{F(\vec{x}^{j-}) - F(\vec{x})}{\sum_{i \in I} w_{i,j}} \tag{3.22}$$

where $\vec{x}^{j-}$ is different from $\vec{x}$ only by item j, $x_i^{j-} = x_i$ for all $i \neq j$ and $x_j^{j-} = 0$, and $F(\vec{x})$ is the fitness value of $\vec{x}$. There are two approaches, the weighted sum

approach (F^{ws}) and the Tchebycheff approach (F^{te}), to determine the fitness value of $\vec{x}$ [121].

$$F^{ws}\left(\vec{x}\middle|\vec{\lambda},\vec{z}\right) = \sum_{i=1}^{m} \lambda_i . \left(z_i - f_i\left(\vec{x}\right)\right) \tag{3.23}$$

$$F^{te}\left(\vec{x}\middle|\vec{\lambda},\vec{z}\right) = \max_{1\leq i\leq m} \left\{\lambda_i . \left|z_i - f_i(\vec{x})\right|\right\} \tag{3.24}$$

where $\vec{z} = (z_1, z_2, \ldots, z_m)$ is the reference point (the least upper bound in the objective space) with respect to the current population, $z_i = \max\left\{f_i\left(\vec{x}\right)|\vec{x} \in \boldsymbol{P}\right\}$, and $\vec{\lambda}$ is the weight vector. There could be several approaches to define $\vec{\lambda}$. Jaszkiewicz [72] suggested $\vec{\lambda}$ to be a random normalised weight vector which is generated in every iteration (within the framework of EMO algorithms). However, the approach proposed by Zhang and Li [121] to define $\vec{\lambda}$ is employed here. Zhang and Li predefine a set of N evenly spread weight vectors $\vec{\lambda}$, controlled by a parameter H, before the start of the evolutionary search. Each vector is the size of m (the number of objectives in the problem). More precisely, $\vec{\lambda^1}, \vec{\lambda^2}, \ldots, \vec{\lambda^N}$ are all the weight vectors in which each individual weight takes a value from:

$$\left\{\frac{0}{H}, \frac{1}{H}, \ldots, \frac{H}{H}\right\} \tag{3.25}$$

Therefore, the number of such vectors is $N = C_{H+m-1}^{m-1} = \frac{(H+m-1)!}{(m-1)!(H+m-1-(m-1))!} = \frac{(H+m-1)!}{(m-1)!H!}$. Different values of H are used for each instance of the 0/1 multiple knapsack problem. For more detail, see [121].

This section reviews 4 different methods for solution representation and infeasible solution repair mechanism for the 0/1 multiple knapsack problem. Different methods tend to have different performance [35, 95]. Therefore, it is suggested that the same methods should be used to eliminate or reduce the bias of one method over the others, when assessing the performance of different EMO algorithms.

It is noted that mainly these EMO algorithms maintain diversity in the objective space.

3.3.2 Performance Metrics

There are several metrics that have been proposed to measure the performance of EMO algorithms. Some of the most popular performance metrics are described below:

Size of the space covered ($\mathcal{S}$). The $\mathcal{S}$-metric [125], also known as the hypervolume metric, measures the size of the region which is dominated by the obtained Pareto front. Therefore the higher value of the $\mathcal{S}$-metric is preferred. In low dimension, 2- and 3- objective spaces, it is known as area and volume respectively. The $\mathcal{S}$-metric requires a reference point which must be dominated by all solutions of the Pareto front. The choice of the reference point is vital to the calculation and assessment of the $\mathcal{S}$-metric. If the reference point is too far away from obtained Pareto sets, it could lead to a considerably high value for $\mathcal{S}$-metric. Consequently, an insignificant difference between obtained Pareto sets makes it difficult to assess the performance of EMO algorithms (figure 3.6).

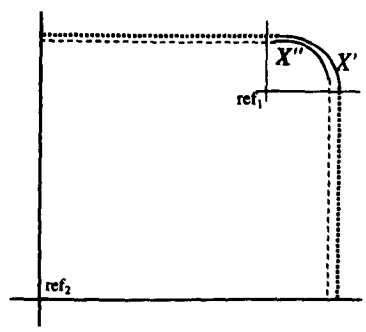


Figure 3.6: The choices of the reference point ref_1 (close to Pareto sets) and ref_2 (at the origin) impose different performance assessment of Pareto sets X' and X'' regarding the $\mathcal{S}$-metric.

It is common that the $\mathcal{S}$-metric is normalised to the percentage point by dividing $\mathcal{S}$-metric by the normal value. This normal value is size of the region form by the reference point (used for the $\mathcal{S}$-metric calculation) and a point that dominates all solutions in the Pareto sets. As aforementioned in section 3.1, it is assumed that the MOP is a maximisation problem. The $\mathcal{S}$-metric calculation is still valid

for minimisation problems or problems with both maximised and minimised objectives. All minimised objectives could be transformed to maximised objectives, then the $\mathcal{S}$-metric calculation could be performed in a uniform manner. The advantage of the $\mathcal{S}$-metric is that it could assess the overall performance of the EMO algorithms without other additional metrics. However, the reference point used for the $\mathcal{S}$-metric calculation must be chosen carefully to avoid ill-assessment. The process to choose this reference point will be further discussed in section 6.2.2.2.

Coverage of two sets ($\mathcal{C}$). The $\mathcal{C}$-metric [125] measures the 'degree' of dominance of a Pareto front over another Pareto front.

$$\mathcal{C}(X', X'') = \frac{|\{x'' \in X''; \exists x' \in X' : x' \succeq x''\}|}{|X''|} \tag{3.26}$$

$\mathcal{C}(X', X'') = 1$ means that all solutions in X'' are either dominated or equal to at least a solution in X'. Therefore the higher value of $\mathcal{C}$-metric is preferred. Both $\mathcal{C}(X', X'')$ and $\mathcal{C}(X'', X')$ have to be considered as it does not always hold that $\mathcal{C}(X', X'') = 1 - \mathcal{C}(X'', X')$.

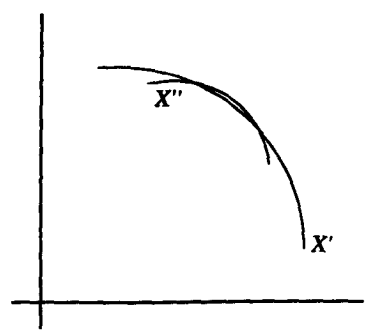


Figure 3.7: X' is better than X'' but X' and X'' are similar under the $\mathcal{C}$-metric.

This metric could be very useful if one Pareto set clearly dominates the other Pareto set. However it does not indicate how much one set is better than the other even in the above case. This metric sometimes misinterpret the performance of EMO algorithms if both $\mathcal{C}(X', X'')$ and $\mathcal{C}(X'', X')$ are low and quite the same. It could imply that both set are quite similar but actually one set is better than the

other (under other performance metric such as the size of the space covered $\mathcal{S}$). This disadvantage is illustrated in figure 3.7.

Generational Distance. The generational distance measures the distance from a non-dominated solution set $\boldsymbol{P}$ to the Pareto-optimal front $\boldsymbol{P}_t$.

$$gd(\boldsymbol{P}, \boldsymbol{P}_t) = \frac{\sqrt{\sum_{\vec{x} \in \boldsymbol{P}} \left(d_{\min}(\vec{x}, \boldsymbol{P}_t)\right)^2}}{|\boldsymbol{P}|} \tag{3.27}$$

where $d_{\min}(\vec{x}, \boldsymbol{P}_t)$ is the minimum Euclidean distance between $\vec{x}$ and points in $\boldsymbol{P}_t$

$$d_{\min}(\vec{x}, \boldsymbol{P}_t) = \min_{\vec{x^*} \in \boldsymbol{P}_t} \left\{ \sqrt{\sum_{i=1}^{m} \left(f_i(\vec{x}) - f_i(\vec{x^*})\right)^2} \right\} \tag{3.28}$$

The generational distance measurement indicates how close a non-dominated solutions set is to the Pareto-optimal front. In other words, this metric indicates the 'degree' of convergence of a non-dominated solution set to any particular part of the Pareto-optimal front. The lower the value of the generational distance, the closer the non-dominated solution set is to a particular part of the true Pareto front indicating better performance of the algorithm. The value $gd(\boldsymbol{P}, \boldsymbol{P}_t) = 0$ indicates that all solutions in $\boldsymbol{P}$ are Pareto-optimal solutions.

Inverted Generational Distance. The inverted generational distance works on the opposite manner to the generational distance metric. It measures the diversity of a non-dominated solutions set along the whole true Pareto front. In other words, this metric indicates how close the Pareto-optimal front is to a non-dominated solutions set is. The lower the value of the inverted generational distance, the more diversity in the non-dominated solution set indicating the better performance of the algorithm. The value $igd(\boldsymbol{P}, \boldsymbol{P}_t) = 1$ means that $\boldsymbol{P}$ is the Pareto optimal set.

$$igd(\boldsymbol{P}, \boldsymbol{P}_t) = \frac{\sqrt{\sum_{\vec{x} \in \boldsymbol{P}_t} \left(d_{\min}(\vec{x}, \boldsymbol{P})\right)^2}}{|\boldsymbol{P}_t|} \tag{3.29}$$

Distance to the Pareto-optimal front ($d(\boldsymbol{P}, \boldsymbol{P}_t)$). The $d(\boldsymbol{P}, \boldsymbol{P}_t)$ measures the average distance of solution in $\boldsymbol{P}$ to the Pareto optimal front $\boldsymbol{P}_t$). The lower value of this metric is preferred.

$$d(\boldsymbol{P}, \boldsymbol{P}_t) = \frac{\sum_{\vec{x} \in \boldsymbol{P}} d_{\min}(\vec{x}, \boldsymbol{P}_t)}{|\boldsymbol{P}|} \tag{3.30}$$

The above metrics measure the diversity, the convergence or the distance to the Pareto-optimal front of obtained Pareto sets. However none of them measure the distribution of solutions in a Pareto set. They do not indicate how close (or far) from one solution to others in the Pareto set is. The next two performance metrics attempt to provide such information of a Pareto set.

Cluster (CL_μ). Wu and Azarm [119] proposed a cluster metric which measures the average number of solutions in each small in grid which size is defined by $\frac{1}{\mu}$, an integer. The objective space is divided into $\frac{1}{\mu}^m$ number of small grids.

$$CL_\mu(\boldsymbol{P}) = \frac{|\boldsymbol{P}|}{NDC_\mu(\boldsymbol{P})} \tag{3.31}$$

where $NDC_\mu(\boldsymbol{P})$ is the number of grids that have at least one solution in $\boldsymbol{P}$. The higher the value of the cluster quantity $CL_\mu(\boldsymbol{P})$ is, the more clustered the solution set is, and the less preferred the solution set. The ideal case is $CL_\mu(\boldsymbol{P}) = 1$ which means all solutions in $\boldsymbol{P}$ belong to different grids.

Schott Spacing (SS). Schott [108] proposed the spacing metric which measure the variance of distance of each solution in $\boldsymbol{P}$ to its closest neighbour:

$$SS(\boldsymbol{P}) = \sqrt{\frac{1}{|\boldsymbol{P}| - 1} \sum_{\vec{x} \in \boldsymbol{P}} \left(\bar{d} - d_{\vec{x}}\right)^2} \tag{3.32}$$

$$d_{\vec{x}} = \min_{\vec{x^*} \neq \vec{x}}^{\vec{x^*} \in \boldsymbol{P}} \left\{ \sum_{l=1}^{m} \left| f_l(\vec{x}) - f_l(\vec{x^*}) \right| \right\} \tag{3.33}$$

$$\bar{d} = \frac{1}{|P|} \sum_{\vec{x} \in P} d_{\vec{x}} \tag{3.34}$$

The lower variance is preferred as it indicate the better distribution of solutions in the Pareto set. The idea value of 0 as it indicates that the distances from one solution to its closest neighbour are the same for every solution in the Pareto set which means a uniform distribution of solutions in the Pareto set.

It is noted that several performance metrics, which require the true Pareto front $\boldsymbol{P}_t$) for calculation, include the generational distance, the inverted generational distance and the distance to the Pareto optimal front. However, it is often true that the Pareto front is unknown for a given problem. In this case, an estimated true Pareto front for that problem could always be used. The estimated true Pareto front could be obtained by using integer programming to solve the problem (with long computational time), solving a relaxed version of the problem, or in the worst case by combining the best solutions from several runs obtained by all algorithms under investigation.

It is usually the case that a number of performance metrics are used in conjunction to assess the performance of EMO algorithms. The reason behind it is that within the multiobjective optimisation framework there are several criteria to assess the EMO algorithms performance such as diversity, convergence and distribution of an obtained set of solutions. One performance metric is often only able to assess on of such criteria. Further more the criteria are normally conflicted. In the author's opinion, amongst researchers the most popular performance metrics, the size of the space covered ($\mathcal{S}$-metric) or the hypervolume, is better than other measurements. The space covered ($\mathcal{S}$-metric) measures the size of the dominated region covered by obtained Pareto sets. This metric could assess both the diversity and the convergence of obtained Pareto sets.

Chapter 4

Restricted Assortative Mating in EMO algorithms

This chapter discusses the role of restricted mating schemes in the context of EMO algorithms. Afterwards, it suggests a dynamic assortative mating scheme that uses similarity in the decision space (genotypic assortative mating) and adapts the mating pressure as the search progresses. The results show that this mechanism improves the performance of SEAMO2 [96] on the multiple 0/1 knapsack problem.

4.1 Introduction

Selection plays an important role within EAs in selecting individuals for survival and selecting parents for recombination. This chapter draws attention to *mating schemes*, i.e. the selection of parents for recombination within EMO algorithms. The concept of restricted mating is quite similar to crowding/clustering techniques deployed by several EMO algorithms. Crowding/clustering techniques select parents for survival aiming to preserve the population's diversity. The restricted mating could preserve the diversity through the avoidance of certain parents re-

combination [33]. Deb and Goldberg [42] also suggested that the recombination of parents from different regions (of both decision and objective space) should be avoided because this type of recombination often creates low quality offspring. There have been a number of mating schemes for single objective EAs proposed in the literature including: fitness proportionate selection, tournament selection, rank-based selection, ancestry selection and assortative mating among others. In *fitness proportionate selection*, parents are chosen based on a probability proportional to their fitness compared to the rest of the population. In *tournament selection*, a group of individuals (usually two) is chosen (usually uniformly at random) from the population and the fittest individual from this group is selected as parent. In *rank-based selection*, individuals are first sorted according to some criteria (usually fitness) and a mapping function is used to assign a selection probability to each individual according to its rank in the ordering. In *ancestry selection* individuals are organised in clans and parents are usually selected from different clans. In *assortative mating* (inspired on natural genetics), individuals are selected based on their similarity (in the objective or the decision space) based on the assumption that recombining parents that 'look' alike produces better offspring. Some mating schemes incorporate some form of *restricted mating* (proposed by Goldberg [55]) where recombination is allowed only if parents meet certain criteria. For reviews on mating schemes and their performance of single-objective evolutionary algorithms see [13, 56, 62].

Despite the various restricted mating schemes that have been investigated for single-objective evolutionary algorithms, the emphasis within EMO algorithms has been mainly on mechanisms to select individuals for survival. In Pareto-based multiobjective optimisation the goal is to find a set of nondominated solutions that is as close as possible to the Pareto optimal front and also well spread and distributed over the trade-off surface [33, 44]. Therefore, most modern EMO algorithms incorporate selection mechanisms, like density-based selection and rank-based selection, in combination with elitism and archiving strategies to ensure the survival of good

nondominated solutions [33, 44, 83]. Also, most EMO algorithms use tournament or other basic selection mechanism for choosing parents and in most cases selection is based on fitness. Some restricted mating schemes have been investigated in the context of EMO algorithms but to a lesser extent than for single-objective evolutionary algorithms. In their book, Coello Coello et al. [33] express that restricted mating has not been fully investigated for EMO algorithms. They also note that there is no conclusive evidence to support whether restricted mating is beneficial or detrimental for the performance of these algorithms. Coello Coello et al. also suggest that experiments investigating the issue of restricted mating should benefit the literature on EMO algorithms.

This chapter proposes a *dynamic assortative mating scheme* [40] for EMO. That is, parents are chosen based on their dissimilarity in the decision space and the dissimilarity threshold or mating pressure σ_{mating} is adapted during the search. The chapter is organised as follows: Section 4.2 gives a more detailed account of related work. Section 4.3 describes the proposed mating scheme and how it is incorporated into SEAMO2. Section 4.3 also presents the experimental setting and results.

4.2 Mating Schemes for EMOAs

This section focuses on mating schemes proposed recently and that restrict mating based on similarity, i.e. assortative mating. For an overview of previous mating schemes within EMOAs refer to the book by Coello Coello et al. ([33], p. 316). Restricted mating has been usually implemented using the σ_{mating} parameter which defines a mating radius or similarity threshold, which can be perceived as the *mating pressure*. Individuals are not allowed to mate if the distance between them (objective or decision space) is larger than σ_{mating}. Kim et al. [76] incorporated neighbourhood crossover into SPEA2 to rank individuals according to how close

they are in the objective space and used binary tournaments to select parents. Few years ago, Ishibuchi and Shibata started an investigation into the effect of restricted mating on the performance of the well-known NSGA2 and SPEA algorithms. In 2003 they proposed a restricted mating scheme based on the similarity between parents (assortative mating) [66]. Later, they modified their approach by incorporating a second layer to select parent A [67]. Their restricted mating scheme works as follows:

1. A set S_A of α candidates is chosen from the current population using iterative binary tournaments.

2. The centre vector (average solution) $\bar{f}(\vec{x}) = (\bar{f}_1(\vec{x}), \bar{f}_2(\vec{x})...\bar{f}_m(\vec{x}))$ in the objective space in S_A is calculated where $\bar{f}_i(\vec{x}) = 1/\alpha \sum_{j=1}^{\alpha} f_i(x_j)$ for $i = 1, 2, ..., m$.

3. The solution in S_A that is most dissimilar (in the objective space) to $\bar{f}(x)$ is chosen as parent A.

4. A set S_B of β candidates is chosen using iterative binary tournaments.

5. The solution in S_B that is most similar to parent A (in the objective space) is chosen as parent B.

Ishibuchi and Shibata [67] observed that their modified mechanism was capable of improving both convergence and diversity in SPEA and NSGA2. However, they also noted that the parameters α and β needed to be carefully adjusted to strike the balance between diversity and convergence speed. Note also that Ishibuchi and Shibata [67] used similarity in the objective space only. In 2004, they reported further experiments to investigate the effect of the mating pressure parameters (α and β) and also the effect of similarity (in the objective space) when selecting parents A and B [68]. They tried their restricted mating mechanism in a number of operation modes resulting from combining different settings: $\alpha = \{1, 2, 3, \ldots\}$;

$\beta = \{1, 2, 3, \ldots\}$; parent A being similar or dissimilar to $\bar{f}(\vec{x})$; parent B being similar or dissimilar to parent A. Once again, they observed that convergence speed and diversity were affected by the settings of α and β. They also expressed that there is a need to set α and β automatically in their mating scheme. More recently, Ishibuchi and Narukawa reported yet more experiments in which they observed that recombining similar parents (which is controlled by varying β) had a positive impact on the performance of NSGA2, although they also observed that recombination seems to be less important than mutation on that particular algorithm [65]. They considered similarity in the objective and the decision space but only when selecting parent B and observed no significant difference in their results [65].

In summary, the investigations by Ishibuchi and Shibata considered fitness-based binary tournaments and distance in the objective space to choose parent A. For selecting parent B, they employed fitness-based binary tournaments and distance both in the objective space and the decision space. The mating pressure is controlled by the number of tournaments (α and β) and by the target similarity to select parent A (with respect to the center vector $\bar{f}(\vec{x})$) and parent B (with respect to parent A). Their results have shown that although their mating scheme is able to improve the performance of SPEA and NSGA2, careful adjustment of the parameters is required to strike the balance between convergence and diversity according to the problem size.

4.3 The Assortative Mating Scheme

The proposed mating scheme does not use tournaments, it uses dissimilarity in the decision space and changes the mating pressure σ_{mating} as the search progresses. Therefore, this scheme differs from those proposed by Kim et al. [76] and Ishibuchi and Shibata [68]. In the proposed *assortative mating scheme* two individuals are

considered for mating only if their dissimilarity (between their gene structures) is above a threshold σ_{mating}. In other words, the two mated parents must be at a certain distance way from each other in the decision space.

4.3.1 The Experimental Setting

The proposed assortative mating scheme is incorporated into the SEAMO2 algorithm [96] and experiments are performed on the multiple 0/1 knapsack problem. SEAMO2 is chosen because it is a simple evolutionary algorithm for multiobjective optimisation that relies mainly on its replacement strategy and it was shown to outperform more elaborate EMO algorithms like NSGA2 and SPEA2 on the multiple 0/1 knapsack problem [96]. Within this chapter, SEAMO2(RM) is referred to the SEAMO2 approach using the proposed scheme for restricted mating. Then, the experiments focus on comparing the performance of SEAMO2(RM) against SEAMO2 [96], SPEA2 [124], NSGA2 [47] and SEAMO2(I) (the SEAMO2 algorithm using Ishibuchi and Shibata's mating strategy [67]) on the multiple 0/1 knapsack problem, with 3 and 4 knapsacks (with population size of 250, 300 and 350 respectively) and 750 items proposed in [125]. Results for short, medium and long runs of 500, 960 and 1920 generations respectively, are reported to investigate the performance of SEAMO2(RM). Results from 30 independent runs for each experiment are used for statistical analysis and discussion. The two metrics, the *size of the space covered* S and the *coverage of two sets* C are used to assess the performance of the restricted mating scheme.

4.3.2 Similarity Measurement

Let's say T and T^* are the sets of packed items in the knapsacks of solutions $\vec{x}$ and $\vec{x^*}$ respectively. Then, the similarity in the decision space between $\vec{x}$ and $\vec{x^*}$, based on the *Jaccard* similarity coefficient (also known as *Jaccard* index), is measured as

follows:

$$sim(\vec{x}, \vec{x^*}) = J(T, T^*) = \frac{|T \bigcap T^*|}{|T \bigcup T^*|} \tag{4.1}$$

For example, $\vec{x} = \{1, 2, 5\}$ and $\vec{x} = \{2, 3, 4, 5\}$, then

$$sim(\vec{x}, \vec{x^*}) = \frac{|\{1,2,5\} \bigcap \{2,3,4,5\}|}{|\{1,2,5\} \bigcup \{2,3,4,5\}|} = \frac{|\{2,5\}|}{|\{1,2,3,4,5\}|} = \frac{2}{5} = 0.4$$

The dissimilarity or the difference in the decision space between $\vec{x}$ and $\vec{x^*}$ is then:

$$diff(\vec{x}, \vec{x^*}) = J_\delta(T, T^*) = 1 - \frac{|T \bigcap T^*|}{|T \bigcup T^*|} \tag{4.2}$$

The recombination of individuals $\vec{x}$ and $\vec{x^*}$ is allowed if and only if $diff(\vec{x}, \vec{x^*}) \geq \sigma_{mating}$ (where $0 \leq \sigma_{mating} \leq 1$). Setting correctly the value of the mating pressure σ_{mating} is important and is discussed in the following sections.

Note that the above definition of similarity/difference is not affected by the solution representation used for the multiple 0/1 knapsack problem. Although the above definition of dissimilarity could be only valid for the multiple 0/1 knapsack problem, the proposed mating scheme can still be implemented if the similarity/difference between two solutions for the problem in hand is measured as a percentage. Therefore, it is argued here that the generality of the proposed mating scheme is not affected by the encoding of solutions or the method used to measure similarity/difference.

4.3.3 Static Setting of the Mating Pressure

The static setting is a simple strategy that presets σ_{mating} before starting the search and this value remains unchanged throughout the evolutionary process. It is required to calculate the value of $diff(\vec{x}, \vec{x^*})$ for every pair of individuals $\vec{x}$ and $\vec{x^*}$ in the population. The σ_{mating} is then set to a value in the *range* of the minimum and maximum values of $diff(\vec{x}, \vec{x^*})$ for every pair of individuals $\vec{x}$ and

$\vec{x^*}$ in the population, where $\vec{x} \neq \vec{x^*}$. The reason is that if σ_{mating} is set to a value smaller than $\min(diff(\vec{x}, \vec{x^*}))$ the restricted mating selection of parents becomes uniform selection, whereas if σ_{mating} is set to a value greater than $\max(diff(\vec{x}, \vec{x^*}))$ no pair of individuals $\vec{x}$ and $\vec{x^*}$ would satisfy the restricted mating condition for recombination.

In order to set σ_{mating} to an appropriate value within this *range* it is possible to let the population evolve for a limited number of generations and observe the trend on the values of $diff(\vec{x}, \vec{x^*})$ in the whole population. A simple experiment on the multiple 0/1 knapsack problem is carried out that allows the population to evolve for 100 generations and the *range* is recorded in every generation. It is observed that the *range* reduces significantly from 60%-70% in the first few generations to 0%-35% in later generations. Therefore, the value of σ_{mating} is set to a value in the range of $(0.0, 0.3)$. Eleven different values of σ_{mating} : $0.050, 0.075, \ldots, 0.300$ are examined in SEAMO2(RM). Results from 30 independent runs are reported in Figure 4.1. Note that the figure only show results for six values of σ_{mating} which are representative of all experimental data. The box-plots in Figure 4.1 correspond to the percentage of the complement of the hyptervolume $\mathcal{S}$ metric, i.e. smaller values indicate better algorithm performance. One box-plot is given for each algorithm: NSGA2, SPEA2, SEAMO2, and SEAMO2(RM) using different values of σ_{mating} which are indicated by NS2, SP2 and SE2 respectively while S.xx indicates SEAMO2(RM) with a given value for σ_{mating}. Results are given for 2- (graphs a-b), 3- (graphs c-d) and 4- (graphs e-f) knapsacks with runs of 500 and 1920 generations.

Figure 4.1 shows clearly that with respect to the *size of the space covered* $\mathcal{S}$ the proposed mating scheme has a positive effect on the performance of SEAMO2(RM). In general, we can see that the performance of SEAMO2(RM) using a preset value of σ_{mating} is consistent over the 30 independent runs (size of the boxplot). There is a significant improvement by applying a higher mating pressure (i.e. increas-

ing the value of σ_{mating}). However, we can also observe that there is an upper limit for the mating pressure after which SEAMO2(RM) starts to perform worse. We can see in Figure 4.1 that this upper limit is about 25% for the 2-knapsack problem (Figure 4.1(a) and 4.1(b)), between 25%-30% for the 3-knapsack problem (Figure 4.1(c) and 4.1(d)), and slighly above 30% for the 4-knapsack problem (Figure 4.1(e) and 4.1(f)). This is simply because when σ_{mating} goes above a given value, no parents can be found that satisfy the restricted mating condition.

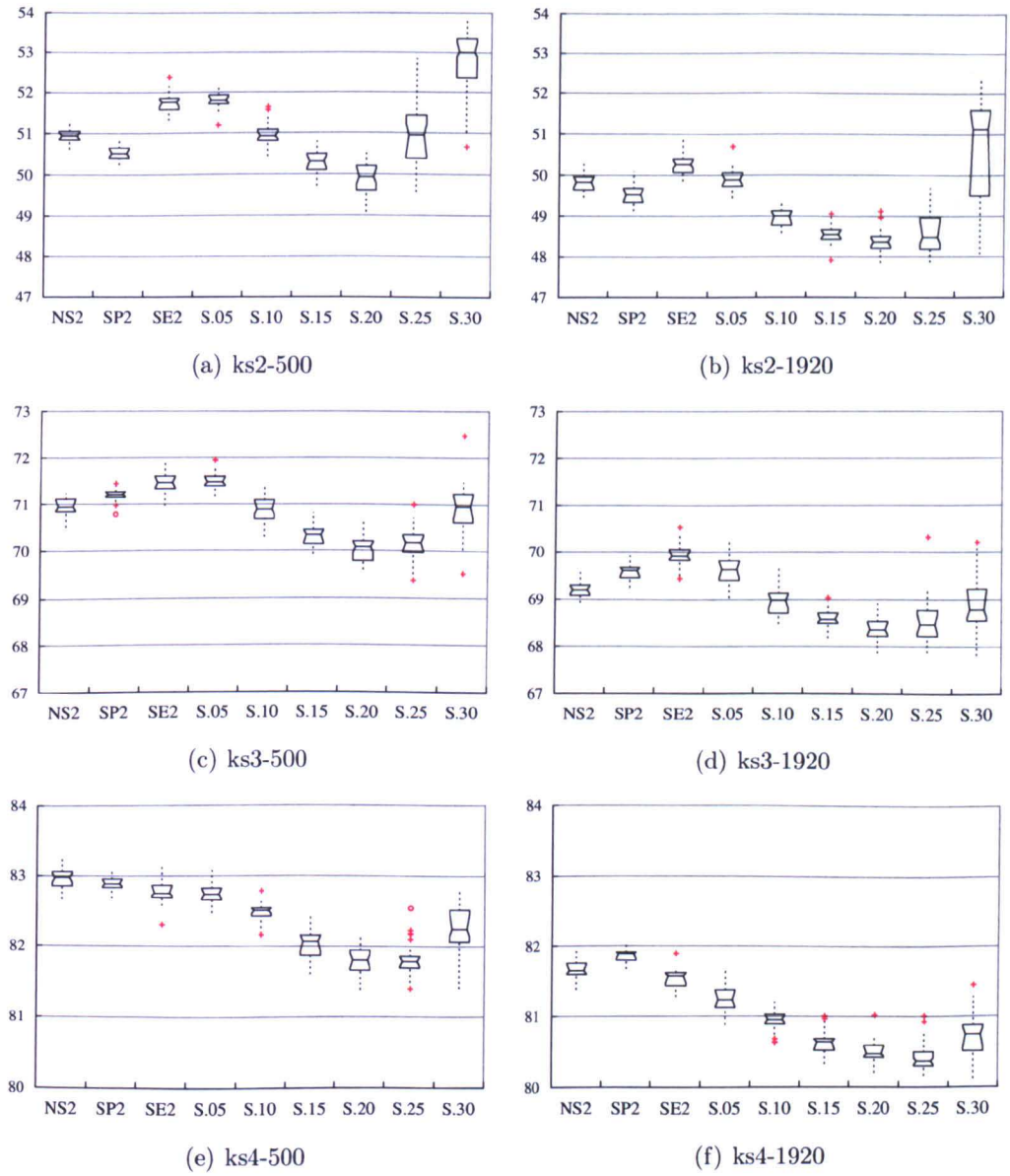


Figure 4.1: Performance of various EMO algorithms on the multiple 0/1 knapsack problem with respect to percentage of the complement of the $\mathcal{S}$ metric.

The results from the combined non-dominated fronts (Figure 4.2) after 30 independent runs of SEAMO2(RM) on the 2-knapsack problem suggest that increasing σ_{mating} seems to have a negative impact on convergence and a positive impact on diversity. For better visualisation, the non-dominated fronts are showed in a lower density (only solutions separated by a distance of at least 400 units in the objective space). The horizontal axis represents profit in knapsack one and the vertical axis represents profit in knapsack two. It is clearly that higher σ_{mating} values reduce the convergence of SEAMO2(RM) but increase diversity (this is similar to the observations by Ishibuchi and Shibata [66]). Therefore, adapting the mating pressure as the evolutionary search progresses is proposed next.

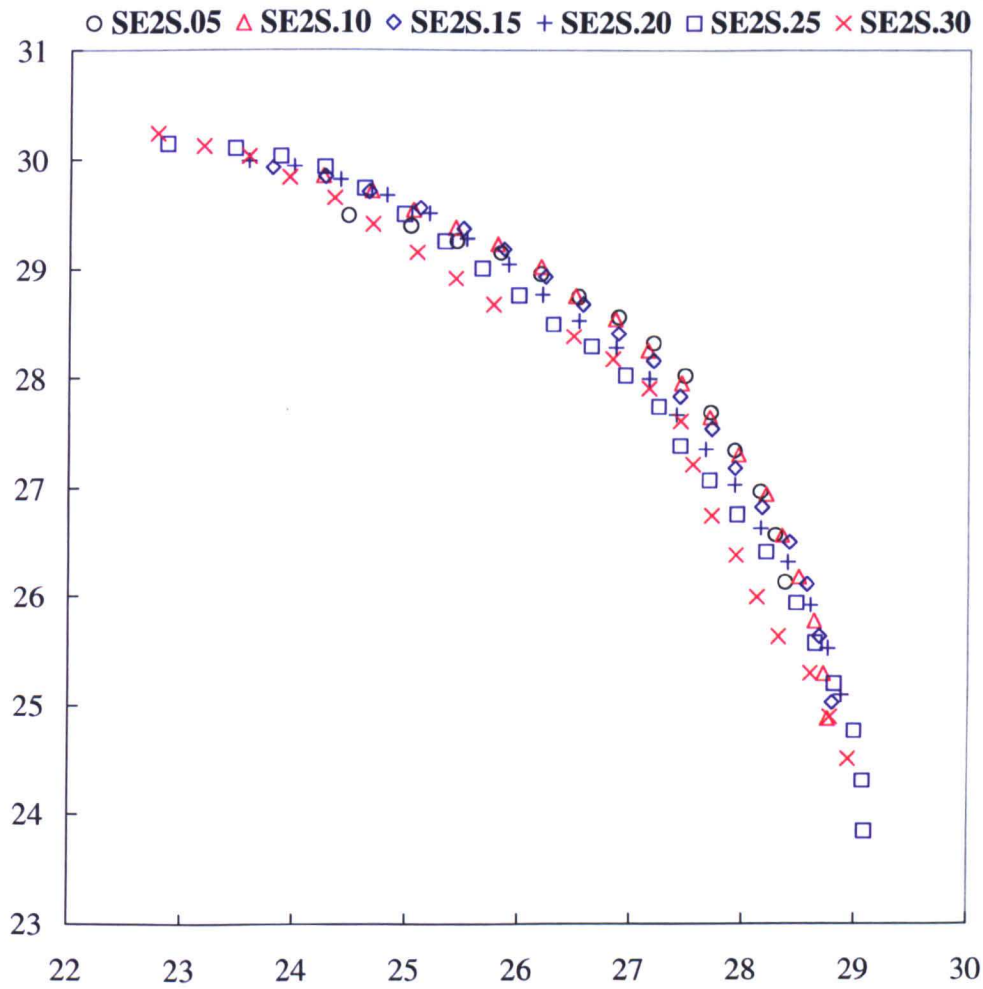


Figure 4.2: Results of SEAMO2(RM) on the 2-knapsack and 750 items problem for six values of σ_{mating}.

4.3.4 Dynamic Setting of the Mating Pressure

The dynamic setting attempt to adapt the σ_{mating} during the evolutionary search. This allows to improve both convergence and diversity of the population along with the evolutionary process. To dynamically change the value of σ_{mating}, it is also required to establish the *range* from which the value of σ_{mating} is then selected. As discussed in section 4.3.3, the value of σ_{mating} is selected randomly with uniform distribution in every generation within the 5% and 95% of the *range* of objectives. This prevents the restricted mating becoming uniform selection (if σ_{mating} is too low) or becoming a non-reproduction scheme (if σ_{mating} is too high). Note that the mating pressure σ_{mating} is set in a dynamic manner as the *range* is adjusted after every generation to reflect the change of diversity (in the decision space) in the population. Then, the chosen value of σ_{mating} will adjust as the population diversity changes. For example, in the first few generations, the population is less 'stable' with many randomly generated solutions provoking a high value of σ_{mating} due to the large different between parents. However, once the population is more 'stable', changes in the value of σ_{mating} drive the population to evolve towards improving diversity (wider *range*) or improving convergence (smaller *range*).

As before, 30 independent runs of SEAMO2(RM) using the dynamic σ_{mating} was executed. The 'best results' obtained using Ishibuchi and Shibata's restricted mating strategy [67] and using the static mating strategy of section 4.3.3 are also included for comparison. These 'best results' are based on the average of the $\mathcal{S}$ metric over 30 independent runs based on 90 combinations of values $\alpha = \{1, 3, 4, \ldots, 9, 10\}$ and $\beta = \{1, 2, \ldots, 9, 10\}$ for Ishibuchi and Shibata's strategy and 11 different values of σ_{mating} in the static mating strategy. These 'best results' are indicated as SE2I and SE2S in Figure 4.3 for Ishibuchi and Shibata's strategy in static restricted mating incorporated into SEAMO2. In the figure, SE2D indicates SEAMO2(RM) using the dynamic σ_{mating}. Figure 4.3 compares NSGA2, SPEA2, SEAMO2, SEAMO2 with Ishibuchi and Shibata's mating strategy, SEAMO2 with

the static σ_{mating} setting and SEAMO2 with the dynamic σ_{mating}. The results of these algorithms and variants, with respect to the complement of the hypervolume $\mathcal{S}$ metric are indicated in the Figure by NS2, SP2, SE2, SE2I, SE2S and SE2D respectively. Results are given for 2- (graphs a-c), 3- (graphs d-f) and 4- (graphs g-i) knapsacks with runs of 500, 960 and 1920 generations. Table 4.1 shows the result for the coverage of two sets $\mathcal{C}$ metric.

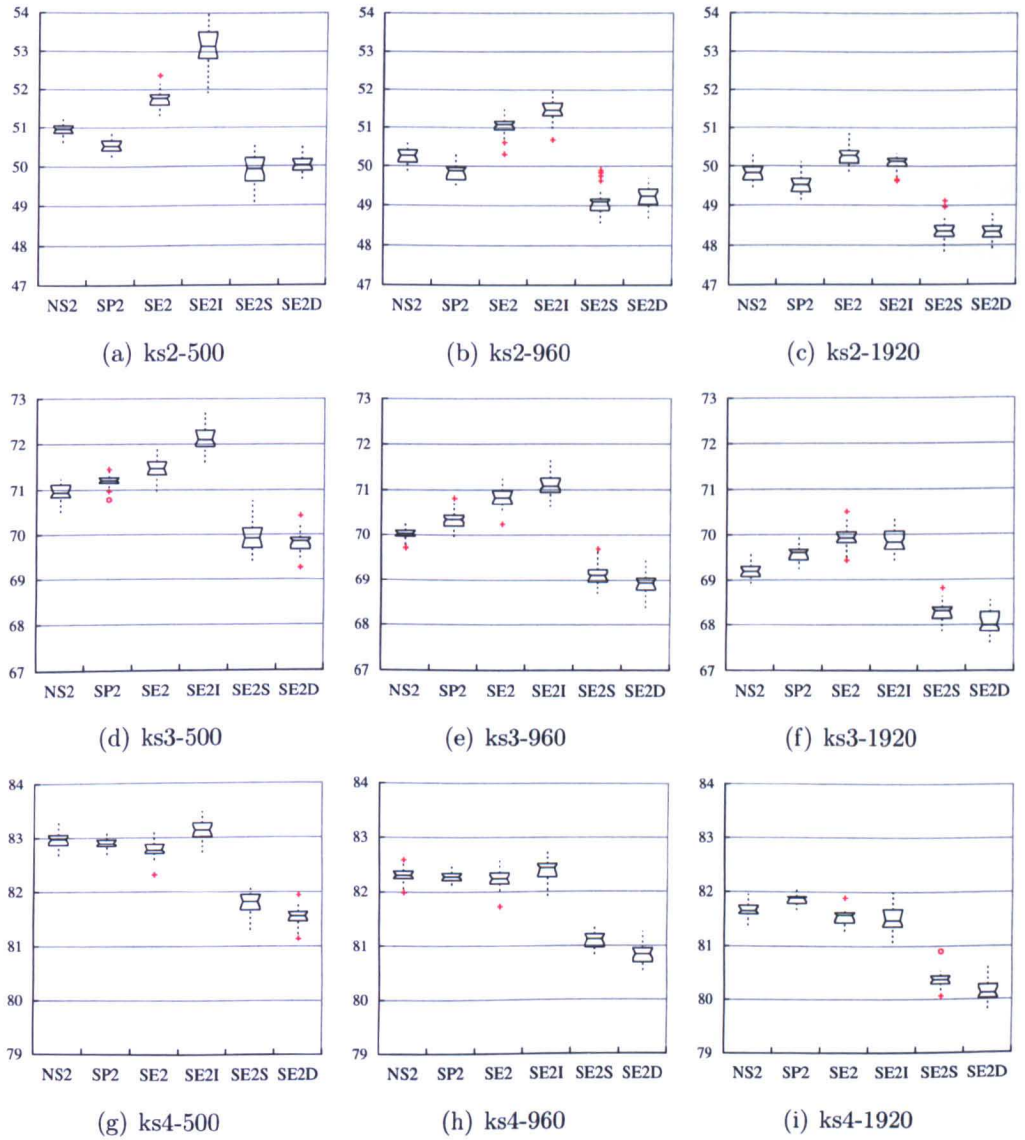


Figure 4.3: Performance of various EMO algorithms on the multiple 0/1 knapsack problem with respect to percentage of the complement of the $\mathcal{S}$ metric.

For each knapsack problem, Figure 4.3 shows the average *non-covered objective space* (smaller values indicate better algorithm performance) at generations 500,

960 and 1920 side by side to facilitate comparison. It is clear that the dynamic setting of σ_{mating} benefits SEAMO2 helping it to outperform NSGA2, SPEA2 and SEAMO2 as well as SEAMO2 with Ishibuchi and Shibata's mating strategy. Furthermore, both static and dynamic mating strategies outperform Ishibuchi and Shibata's restricted mating strategy when incorporated into SEAMO2. In most cases, the dynamic strategy outperforms the static one with the exception of the 2-knapsack problem with short and medium runs (graphs a-b in Figure 4.3). Table 4.1 shows the strong performance of SEAMO2D (the dynamic restricted mating incorporated in SEAMO2) particularly on problems with 3 and 4 knapsacks. From Figure 4.3 and Table 4.1 it could be seen that the dynamic mating strategy significantly improves diversity but it slightly worsens convergence in the higher dimension problem (4 knapsacks). It is also noticed an interesting result is that Ishibuchi and Shibata's strategy seems to worsen the performance of SEAMO2 (it was reported in [67] that Ishibuchi and Shibata's strategy improves the performance of SPEA and NSGA2). This is more noticeable in the early stages of the evolutionary search (generations 500 and 960) in low dimension problems (2 and 3 knapsacks). It is believed that Ishibuchi and Shibata's mating strategy conforms with the selection strategy in SPEA and NSGA2 where individuals are uniformly chosen using tournament selection. However, Ishibuchi and Shibata's mating strategy interferes with the selection strategy in SEAMO2 (Ishibuchi and Shibata's mating strategy chooses the first parent with binary tournaments while in SEAMO2 each individual acts as the first parent once), while the proposed restricted mating does not alter the original behaviour of SEAMO2.

Figure 4.4 shows (in lower density as in Figure 4.2) the combined nondominated fronts over 30 runs on the 2-knapsack problem. Figure 4.4 also shows that SEAMO2(RM) using the dynamic mating strategy outperforms SPEA2 and NSGA2 but its convergence is just slightly lower than for SEAMO2. Overall, results in Figure 4.3, Figure 4.4 and Table 4.1 give evidence that the dynamic setting of σ_{mating} is beneficial for SEAMO2 on the three multiple 0/1 knapsack problems.

Table 4.1: Average values (standard deviation) of *coverage of two sets* $\mathcal{C}(A \succeq B)$.

$\mathcal{C}(A \succeq B)$										
Algorithm		2 knapsacks			3 knapsacks			4 knapsacks		
A	B	500	960	1920	500	960	1920	500	960	1920
NSGA2	SEAMO2D	3(7)	12(16)	24(20)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)
SPEA2		2(3)	8(7)	18(16)	0(0)	0(0)	0(0)	0(0)	0(0)	0(0)
SEAMO2		11(17)	18(20)	26(19)	21(17)	24(16)	26(15)	26(22)	21(18)	19(14)
SEAMO2I		0(0)	0(0)	0(1)	0(0)	0(0)	2(5)	0(0)	0(1)	1(3)
SEAMO2S		2(3)	3(4)	5(5)	0(1)	1(1)	1(1)	0(1)	1(1)	1(1)
SEAMO2D	NSGA2	89(12)	69(24)	46(27)	92(7)	84(7)	77(8)	100(1)	99(3)	98(3)
	SPEA2	89(10)	74(15)	53(22)	88(8)	64(10)	45(9)	95(4)	84(7)	76(7)
	SEAMO2	76(34)	60(38)	47(34)	34(29)	24(25)	18(19)	16(20)	15(18)	12(12)
	SEAMO2I	100(0)	100(3)	92(15)	100(1)	98(2)	82(25)	91(16)	84(23)	65(32)
	SEAMO2S	80(8)	82(11)	83(10)	86(6)	83(6)	78(8)	79(6)	75(7)	67(7)

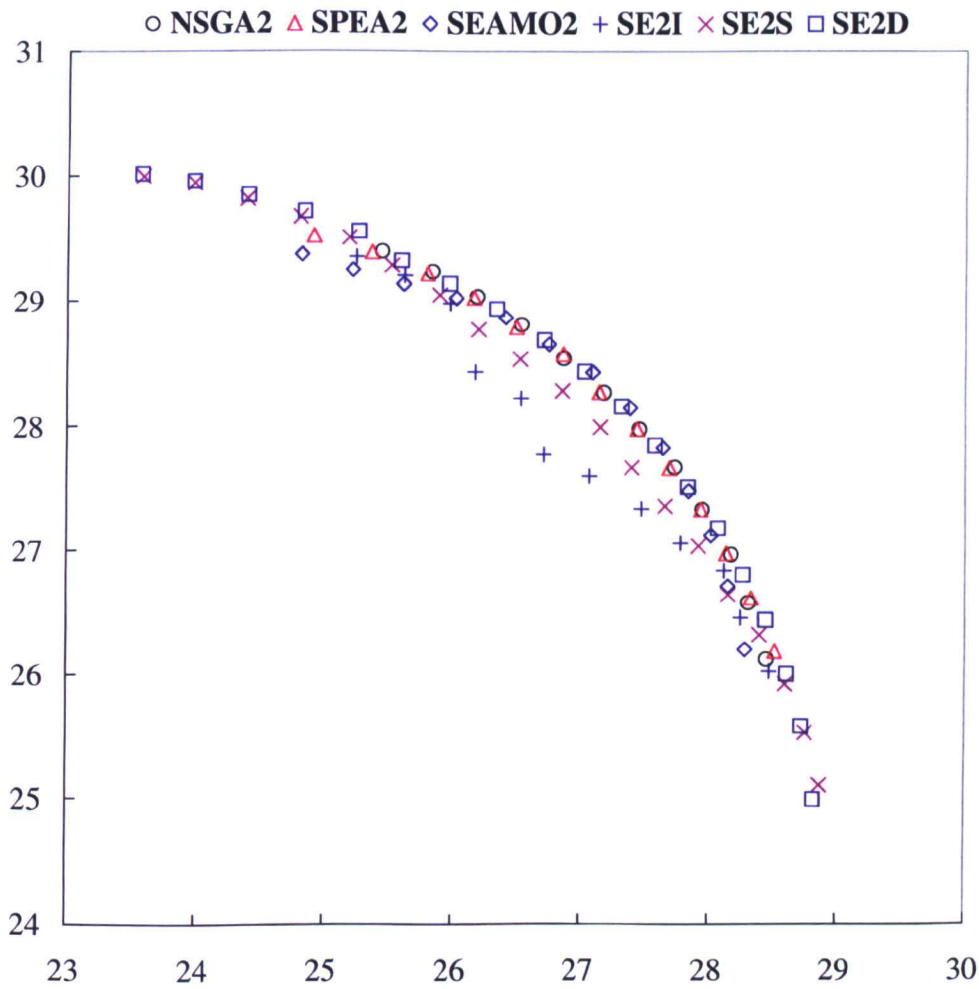


Figure 4.4: Results comparing NSGA2, SPEA2, SEAMO2, SE2I, SE2S and SE2D on the 2-knapsack problem with 750 items.

Figure 4.5 compares the proposed assortative mating scheme using the static setting and using the dynamic setting over 30 independent runs for the 2-knapsack problem with 750 items. The various static settings are indicated by SE2S.xx and the dynamic setting is indicated by SE2D. Figure 4.5 shows that the *dynamic assortative mating scheme* can simultaneously maintain the convergence and the diversity of the population but the static setting can only give a positive effect on the convergence (using lower σ_{mating}) or on the diversity (using higher σ_{mating}) but not both at the same time. This shows that adapting the *diff(range)* (from where σ_{mating} is chosen) according to the population diversity during evolution, helps to strike a balance between convergence and diversity. Of course, more elaborate methods for adapting the mating pressure can be investigated, but the proposed one points us on the right direction.

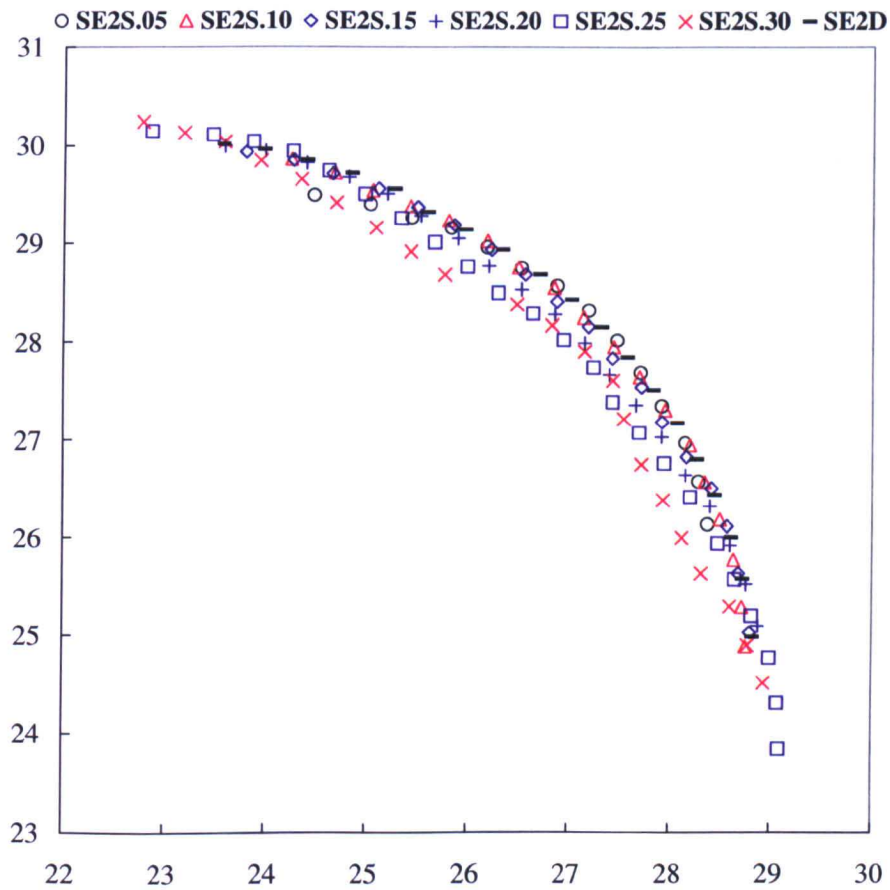


Figure 4.5: Results comparing SE2S and SE2D on the 2-knapsack problem with 750 items.

4.4 Summary

This chapter proposes a restricted mating scheme for evolutionary multiobjective algorithms. This mating scheme is *assortative* because it selects parents based on their similarity in the decision space. Setting the mating pressure σ_{mating} to a constant value provokes either convergence or diversity to be negatively affected. Therefore, the proposed scheme is *dynamic* because it varies σ_{mating} taking into account the population diversity in the decision space. Experiments show that the simple mechanism to adapt the mating pressure helps SEAMO2 (simple evolutionary algorithm for multiobjective optimisation) to improve its performance while striking a good balance between convergence and diversity. The proposed mating scheme can be incorporated into different EMO algorithms because it does not alter their original selection strategy.

The study of the restricted mating scheme in this chapter show that the performance result for the multiple 0/1 knapsack problem using a simple heuristic decoder is improved by mating two different parents (in the decision space). This restricted mating scheme is also further applied to solve a nurse scheduling problem proposed by Landa-Silva and Le [81]. However it does not improve the result for this problem. Further experiments show that the heuristic decoder for this nurse scheduling problem highly affects the search which then eliminates the effect of the restricted mating on the search performance. Chapter 7 will further look into this issue to reduce the effect of the heuristic decoder.

Chapter 5

Volume Dominance in EMO

An important goal in multiobjective optimisation is to find a good set of nondominated solutions that is both well-distributed and well-converged. Most multiobjective optimisation algorithms use the conventional Pareto dominance relationship. Over recent years, new alternative approaches to the conventional Pareto dominance relationship, such as relaxed Pareto dominance, have been proposed which will be reviewed later in this chapter. The interest in relaxed forms of Pareto dominance has been increasing due to their capability to find extreme values in the objective space. First, this chapter conducts a short review on relaxed forms of Pareto dominance in the literature. Then, a new form of relaxed Pareto dominance, called *volume dominance*, is presented. We evaluate the performance of some EMO algorithms when using the proposed *volume dominance* instead of Pareto dominance. The results for this study using SEAMO2, SPEA2 and NSGA2 show that the proposed *volume dominance* is capable of obtaining a better and smoother trade-off front, and it is more robust than Pareto dominance which allows EMO algorithms to obtain more consistent performance.

5.1 Introduction

EAs are capable of generating multiple promising solutions in a single run and evolving a population of solutions towards the Pareto front. These properties make EAs especially adequate to deal with Pareto based multiobjective optimisation problems. A good EMO algorithm should be able to obtain Pareto fronts that are both well-distributed and well-converged. Two issues when designing an EMO algorithm are to decide how solutions in the population should be evolved and to decide how to establish superiority between solutions in the population (i.e. how to compare solution fitness in a multiobjective sense). With respect to the first issue, a number of techniques have been investigated to 'push' solutions towards the desired part of the tradeoff surface. For example, directed weighted vectors, restricted mating, archiving elite solutions, clustering/crowding, fitness sharing, specialised operators, etc. have been proposed to improve the distribution and the convergence of the population towards the Pareto front [33, 46, 80]. However, in our opinion, the latter issue of assigning fitness to solutions in the multiobjective context has received less attention than it deserves.

For assigning fitness to solutions, most modern EMO algorithms adopt the conventional Pareto dominance relationship. There are few papers that propose different types of dominance relationship such as α-dominance, ϵ-dominance, E-Pareto dominance and fuzzy dominance (details in section 5.2). These variations of dominance aim to help in finding solutions in difficult areas (like the extremes of the tradeoff surface) or attempt to combine convergence and diversity in order to achieve a better Pareto front in difficult problems. These variations of Pareto dominance, called *relaxed Pareto dominance* here, apply some transferring functions to the objective values before comparing the solutions using Pareto dominance. It has been shown that *relaxed Pareto dominance* helps to obtain better quality Pareto fronts in some problems (e.g. [82, 79, 27]).

This chapter proposes a new form of relaxed Pareto dominance, called *volume dominance.* The *volume dominance* proposed here establishes the dominance relationship between two solutions in the multiobjective context based on the dominated volume in the objective space. The performance of the proposed *volume dominance* is compared to its counterpart, the conventional Pareto dominance, by deploying both types of dominance in some well-known EMO algorithms. The multiple knapsack problem is used in our experiments because benchmark results are available for this problem.

5.2 Literature Review

The concept of Pareto dominance is briefly discussed in section 3.1 of chapter 3. Although the conventional Pareto dominance has been widely accepted as the main technique to compare the quality of solutions in EMO, there are proposals for different types of dominance relationship as a way to improve the performance of multiobjective optimisers. These variations of Pareto dominance, such as α-dominance, ϵ-dominance, E-Pareto dominance and fuzzy dominance, are known as *relaxed Pareto dominance* and they are slightly different from the conventional Pareto dominance. Relaxed forms of dominance may allow a solution $\vec{x}$ to dominate another solution $\vec{x^*}$ for which $\vec{x}$ and $\vec{x^*}$ are Pareto nondominated solutions or even $\vec{x}$ is Pareto-dominated by $\vec{x^*}$. Some examples of relaxed dominance are described below.

Structure of Domination. The first form of relaxed Pareto dominance was that by Yu in 1974 who proposed a structure of domination over the objective space to explore the geometry of the set of all nondominated solutions [120]. Two new concepts of cone convexity and cone extreme points were introduced to study decision problems on polar cones and polyhedral cones. However, it is not until recently that there is an increasing interest in relaxed Pareto dominance within

the multiobjective optimisation community.

α-Dominance. In 2001, Kokolo et al. introduced α-dominance to deal with what they call *dominance resistant solutions*, i.e. solutions that are fairly inferior qualitatively but for which dominating solutions are scarcely found [63]. The main idea of α-dominance is to set up upper and lower bounds of trade-offs between objectives. In α-dominance, small detriments in one objective are considered acceptable if it leads to a noticeable improvement in other objectives.

ϵ-Dominance. Laumanns et al. proposed a slightly different but simpler form of relaxed dominance called ϵ-dominance which seeks to combine diversity and convergence in one criterion [82]. A solution $\vec{x}$ is said to ϵ-dominate a solution $\vec{x^*}$ if and only if $(1+\epsilon).x_i \geq x_i^* \ \forall i \in 1,\ldots,m$. The main difference between ϵ-dominance and α-dominance is that ϵ-dominance allows some Pareto-dominated solutions (i.e. $\vec{x} \succ \vec{x^*}$) to actually become preferred (i.e. $\vec{x^*}$ ϵ-dominates $\vec{x}$) which is not the case in α-dominance.

Extended Pareto Dominance (E-Pareto). Jin and Wong proposed an extended Pareto dominance (E-Pareto) in their Adaptive Rectangle Archiving algorithm [73]. Extended Pareto dominance is quite similar to ϵ-dominance in the sense that both apply some sort of transferring functions to the objective vector before comparing them. In the extended Pareto dominance, $\vec{x}$ E-dominates $\vec{x^*}$ for some transferring function, FUN, and a constant vector e (> 0) if and only if $\forall i \in 1,\ldots,m \ FUN(f_i(\vec{x})) \geq FUN(f_i(\vec{x^*})) - e_i$, where m is the number of objectives and $f_i(\vec{x})$ is the ith objective function of $\vec{x}$. Jin and Wong compared E-dominance to ϵ-dominance and the conventional Pareto dominance and stated that E-dominance becomes ϵ-dominance as $FUN(f_i(\vec{x})) = ln(f_i(\vec{x}))$ and $e_i = ln(1+\epsilon)$ and Pareto dominance as $FUN(f_i(\vec{x})) = f_i(\vec{x})$ and $e_i = 0$.

Fuzzy Pareto Dominance. Several researchers have investigated the fuzzification of Pareto dominance. In fuzzy-Pareto-dominance, proposed by Koppen et

al. [79], $\vec{x}$ dominates $\vec{x^*}$ by degree μ_a with

$$\mu_a(\vec{x}, \vec{x^*}) = \frac{\prod_{i=1}^{m} min(f_i(\vec{x}), f_i(\vec{x^*}))}{\prod_{i=1}^{m} f_i(\vec{x^*})}$$

and $\vec{x}$ is dominated by $\vec{x^*}$ at degree μ_p with

$$\mu_p(\vec{x}, \vec{x^*}) = \frac{\prod_{i=1}^{m} min(f_i(\vec{x}), f_i(\vec{x^*}))}{\prod_{i=1}^{m} f_i(\vec{x})}$$

This fuzzy form of Pareto dominance becomes the conventional Pareto dominance (for $\vec{x} \succ \vec{x^*}$) when $\mu_a(\vec{x}, \vec{x^*}) = 1$ and $\mu_p(\vec{x^*}, \vec{x}) = 1$, but $\mu_p(\vec{x}, \vec{x^*}) < 1$ and $\mu_a(\vec{x^*}, \vec{x}) < 1$. Koppen et al. applied fuzzy-Pareto-dominance to deal with what they call the Pareto-Box problem, determining the expectation value for the size of the Pareto front of m points in an n-dimensional space.

Peng et al. proposed a different concept for fuzzy dominance based on the credibility distribution of fuzzy variables [102]. Here, ξ and η are two fuzzy variables with the credibility distribution $\Phi(x)$ and $\Psi(x)$ respectively, where $\Phi^{(k)}(x) = \int_{-\infty}^{x} \Phi^{(k-1)}(t)dt$ and $\Psi^{(k)}(x) = \int_{-\infty}^{x} \Psi^{(k-1)}(t)dt$ $k = 2, 3, \ldots$ and $\Phi^{(1)}(x) = \Phi(x)$ and $\Psi^{(1)}(x) = \Psi(x)$. Then, it is said that ξ *k-Order Fuzzy Dominates* η if and only if $\Phi^{(k)}(x) \leq \Psi^{(k)}(x)$ $\forall x \in \Re$.

Gaining Factor. Burke and Landa-Silva proposed another form of relaxed dominance using a *gaining* factor [27]. For a 2-objective maximisation problem, $\vec{x}$ dominates $\vec{x^*}$ if $f_2(\vec{x}).(1 + gain) > f_2(\vec{x^*})$ where $gain = (f_1(\vec{x}) - f_1(\vec{x^*}))/f_1(\vec{x})$ which is equivalent to the following relation:

$$\frac{f_1(\vec{x^*})}{f_1(\vec{x})} + \frac{f_2(\vec{x^*})}{f_2(\vec{x})} < 2$$

For m objectives maximisation problem, $\vec{x}$ dominates $\vec{x^*}$ if

$$\frac{f_1(\vec{x^*})}{f_1(\vec{x})} + \frac{f_2(\vec{x^*})}{f_2(\vec{x})} + \ldots + \frac{f_m(\vec{x^*})}{f_m(\vec{x})} < m$$

They showed that the performance of two multiobjective algorithms was improved by using relaxed dominance when solving a highly constrained combinatorial optimisation problem.

g-dominance. Julian et al. proposed a variation concept of the Pareto dominance, called *g-dominance*, which is based on the information included in a reference point [93]. Unlike other relaxed Pareto dominance, *g-dominance* is only interested in a certain area of the search space which results in only a certain part of the Pareto front. The main aim of *g-dominance* is to approximate the Pareto front around the area defined by the reference point. Given a reference point $\vec{v}$ and a point $\vec{w}$, $Flag_{\vec{v}}(\vec{w})$ is defined as follows:

$$Flag_{\vec{v}}(\vec{w}) = \begin{cases} 1 & \text{if } w_i \leq v_i \forall i = 1, \ldots, m \\ 1 & \text{if } v_i \leq w_i \forall i = 1, \ldots, m \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

Then *g-dominance* is as follows: a solution $\vec{x}$ *g-dominates* a solution $\vec{x^*}$ for a given reference point $\vec{g}$ if

1. $Flag_{\vec{g}}(\vec{x}) > Flag_{\vec{g}}(\vec{x^*})$ or
2. $Flag_{\vec{g}}(\vec{x}) = Flag_{\vec{g}}(\vec{x^*})$ and $\vec{x}$ *Pareto-dominates* $\vec{x^*}$

Equation (5.1) means that $Flag_{\vec{v}}(\vec{w})$ is set to 1 when either $\vec{v}$ Pareto-dominates $\vec{w}$ or vice versa and $Flag_{\vec{v}}(\vec{w})$ is set to 0 if $\vec{v}$ and $\vec{w}$ are Pareto nondominated. In the other word, if a solution $\vec{x}$ is in the preferred area in objective space w.r.t a given reference point $\vec{g}$ then trigger $Flag_{\vec{g}}(\vec{x})$ (set to 1). Therefore, in words, a solution $\vec{x}$ *g-dominates* a solution $\vec{x^*}$ for a given reference point $\vec{g}$ if $\vec{x}$ is in the preferred area defined by $\vec{g}$ but not $\vec{x^*}$, otherwise if both $\vec{x}$ and $\vec{x^*}$ are either or not in the preferred area defined by $\vec{g}$ then $\vec{x}$ *g-dominates* $\vec{x^*}$ if $\vec{x}$ *Pareto-dominates* $\vec{x^*}$. An interactive scheme to modify the reference point $\vec{g}$ is also suggested by Julian et al. [93]. The purpose of *g-dominance* is to drive the search to the preferred

area set by the decision maker. See [93] for more detail on how to efficiently use *g-dominance* in EMO.

5.3 Volume Dominance

All the above forms of relaxed dominance, like the conventional Pareto dominance, are based on comparing the objective vectors of solutions in one way or another. This chapter proposes a new form of relaxed Pareto dominance, called *volume dominance* which is based on a different concept: comparing the dominated volumes in the objective space between two solutions $\vec{x}$ and $\vec{x^*}$. This property makes *volume dominance* distinguishable from conventional Pareto dominance and other relaxed forms of dominance.

The dominated volume of solution $\vec{x}$ is defined as the region R for which all feasible solutions in R are dominated by $\vec{x}$. In order to determine the dominated volume of solution $\vec{x}$, it is required to define a reference point $\vec{r}$ in the objective space corresponding to a solution whose is dominated by all other solutions which objective vectors are also in R. Hence, the formula to calculate the dominated volume of $\vec{x}$ with respect to the reference point $\vec{r} = (r_1, r_2, \ldots, r_m)$ is defined as follows:

$$V_{\vec{x}} = \prod_{i=1}^{m} (f_i(\vec{x}) - r_i) \tag{5.2}$$

Then the dominated volumes of $\vec{x}$ and $\vec{x^*}$ are compared to establish the dominance relationship between $\vec{x}$ and $\vec{x^*}$.

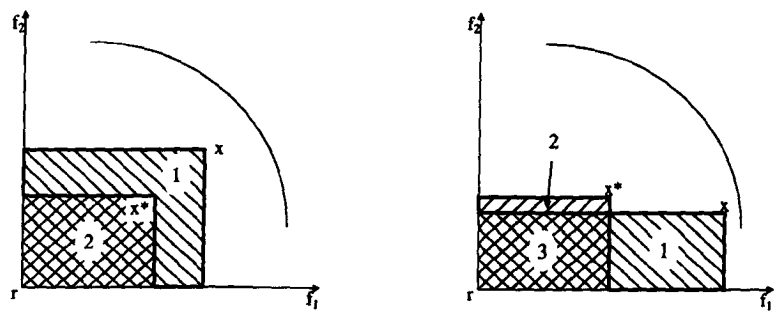
It should be noted that this proposed volume dominance relationship is not based on directly comparing the two dominated volumes. Instead, it is based on the *relative dominated volume.* The relative dominated volume is the volume of the region that is dominated by both $\vec{x}$ and $\vec{x^*}$, called *shared dominated volume.*

The shared dominated volume is defined as follows:

$$SV_{\vec{x},\vec{x^*}} = \prod_{i=1}^{m}(min(f_i(\vec{x}), f_i(\vec{x^*})) - r_i) \tag{5.3}$$

The volume dominance relationship of $\vec{x}$ and $\vec{x^*}$ is then established by comparing the dominated volumes $V_{\vec{x}}$ and $V_{\vec{x^*}}$ to the shared dominated volume $SV_{\vec{x},\vec{x^*}}$. Then, it is said that $\vec{x^*}$ is volume-dominated by $\vec{x}$ ($\vec{x^*} \prec_V \vec{x}$) for some ratios rSV if either:

1. $V_{\vec{x^*}} = SV_{\vec{x},\vec{x^*}}$ and $V_{\vec{x}} > SV_{\vec{x},\vec{x^*}}$ or
2. $V_{\vec{x}} > V_{\vec{x^*}} > SV_{\vec{x},\vec{x^*}}$ and $r_{\vec{x},\vec{x^*}} = \frac{V_{\vec{x}} - V_{\vec{x^*}}}{SV_{\vec{x},\vec{x^*}}} > rSV$



(a) Case 1 (b) Case 2

Figure 5.1: Volume Dominance - A Form of Relaxed Dominance

Figure 5.1(a) illustrates case 1 of the volume dominance. With respect to the reference point r, the dominated volume of $\vec{x^*}$ is indicated as the region 2 which is also the shared dominated volume of $\vec{x}$ and $\vec{x^*}$ ($V_{\vec{x^*}} = SV_{\vec{x},\vec{x^*}}$). The dominated volume of $\vec{x}$ is indicated as the combined region of region 1 and region 2 (hence $V_{\vec{x}} > SV_{\vec{x},\vec{x^*}}$). This is clearly the case of Pareto domination.

Figure 5.1(b) illustrates case 2 of the volume dominance. With respect to the reference point r, the dominated volume of $\vec{x^*}$ is indicated as the combined region of region 2 and region 3. The dominated volume of $\vec{x}$ is indicated as the combined region of region 1 and region 3. The shared dominated volume of $\vec{x}$ and $\vec{x^*}$ is region 3. Hence, $r_{\vec{x},\vec{x^*}}$ is the ratio of the difference between region 1 and region 2 ($V_{\vec{x}} - V_{\vec{x^*}}$) to region 3, the shared dominated volume of $\vec{x}$ and $\vec{x^*}$ ($SV_{\vec{x},\vec{x^*}}$).

The value of the parameter rSV indicates how much larger the unshared volume, the difference of region 1 to region 2 ($V_{\vec{x}} - V_{\vec{x^*}}$), with respect to the shared volume, region 3 ($SV_{\vec{x},\vec{x^*}}$), to allow $\vec{x}$ volume-dominates $\vec{x^*}$. A small value for the parameter rSV indicates that a small difference between the dominated volumes of $\vec{x}$ and $\vec{x^*}$ (w.r.t $SV_{\vec{x},\vec{x^*}}$) is enough to discriminate between $\vec{x}$ and $\vec{x^*}$ (or to allow $\vec{x}$ volume-dominates $\vec{x^*}$).

It is noted that the proposed *volume dominance* contains a normalisation element which prevents bias in some directions in cases with non-commensurable objective functions. *Proof:* Suppose it is required to normalise each objective function using $\vec{n} = \{n_i | i : 1 \ldots m\}$ then

$$
\begin{aligned}
V_{\vec{x}}^N &= \prod_{i=1}^{m} \left(\frac{f_i(\vec{x})}{n_i} - \frac{r_i}{n_i} \right) = \prod_{i=1}^{m} \frac{f_i(\vec{x}) - r_i}{n_i} \\
&= \frac{\prod_{i=1}^{m} (f_i(\vec{x}) - r_i)}{\prod_{i=1}^{m} n_i} = \frac{V_{\vec{x}}}{\prod_{i=1}^{m} n_i} \\
SV_{\vec{x},\vec{x^*}}^N &= \prod_{i=1}^{m} \left(min \left(\frac{f_i(\vec{x})}{n_i}, \frac{f_i(\vec{x^*})}{n_i} \right) - \frac{r_i}{n_i} \right) \\
&= \prod_{i=1}^{m} \frac{min(f_i(\vec{x}), f_i(\vec{x^*})) - r_i}{n_i} \\
&= \frac{\prod_{i=1}^{m} (min(f_i(\vec{x}), f_i(\vec{x^*})) - r_i)}{\prod_{i=1}^{m} n_i} = \frac{SV_{\vec{x},\vec{x^*}}}{\prod_{i=1}^{m} n_i} \\
r_{\vec{x},\vec{x^*}}^N &= \frac{V_{\vec{x}}^N - V_{\vec{x^*}}^N}{SV_{\vec{x},\vec{x^*}}^N} = \frac{\frac{V_{\vec{x}}}{\prod_{i=1}^{m} n_i} - \frac{V_{\vec{x^*}}}{\prod_{i=1}^{m} n_i}}{\frac{SV_{\vec{x},\vec{x^*}}}{\prod_{i=1}^{m} n_i}} = \frac{V_{\vec{x}} - V_{\vec{x^*}}}{SV_{\vec{x},\vec{x^*}}} = r_{\vec{x},\vec{x^*}}
\end{aligned}
$$

The proposed *volume dominance* also covers Pareto dominance (figure 5.1(a)). *Proof:* if $\vec{x}$ Pareto-dominates $\vec{x^*}$ ($\vec{x} \succ \vec{x^*}$) i.e. $f_i(\vec{x}) \geq f_i(\vec{x^*})$ $\forall i = 1, \ldots, m$ and $f_i(\vec{x}) > f_i(\vec{x^*})$ for at least one $i = 1, \ldots, m$. The shared dominated volume:

$$
SV_{\vec{x},\vec{x^*}} = \prod_{i=1}^{m} (min(f_i(\vec{x}), f_i(\vec{x^*})) - r_i) = \prod_{i=1}^{m} (f_i(\vec{x^*}) - r_i) = V_{\vec{x^*}}
$$

As $f_i(\vec{x}) \geq f_i(\vec{x^*})$ $\forall i = 1, \ldots, m$ and $f_i(\vec{x}) > f_i(\vec{x^*})$ for at least one $i = 1, \ldots, m$, then $V_{\vec{x}} > V_{\vec{x^*}}$ i.e. $V_{\vec{x}} > SV_{\vec{x},\vec{x^*}}$. Therefore, $\vec{x}$ volume-dominates $\vec{x^*}$ ($\vec{x} \succ_V \vec{x^*}$).

At first sight, the above volume dominance relationship seems to be very similar to the well-known $\mathcal{S}$-metric (hypervolume) proposed in [125]. However the underlying principle is different. The $\mathcal{S}$-metric measures the size of the volume covered by a set of nondominated solutions to determine how good that set is in comparison to another set of nondominated solutions. The *volume dominance* compares the sizes of volume covered by each solution to establish the dominance relationship between any two solutions. Moreover, when using the $\mathcal{S}$-metric, the volumes covered by the two non-dominated sets are compared directly while in the *volume dominance* the shared dominated volume is also considered.

There is a crucial difference between the proposed *volume dominance* and other dominance relationships including conventional Pareto dominance and forms of relaxed dominance previously proposed in the literature. In order to decide dominance (discriminate) between solutions, *volume dominance* takes all objectives into consideration at once by combining them into a single unit vector rather than directly comparing each objective in turn as it happens in other dominance relationships. This allows *volume dominance* to evaluate the whole objective vector to compensate improvement and detriment between objectives.

5.3.1 Experimental Design

Most relaxed forms of dominance presented in the literature aim to reach and maintain extreme points in the objective space or points that are difficult to reach and maintain with conventional Pareto dominance. Other relaxed forms of dominance aim to combine diversity and convergence into a single criterion when discriminating between solutions. These relaxed dominances have been proposed as an integral part of specific multiobjective algorithms [82, 79, 27, 120, 63, 102]. To the best of our knowledge, none of these forms of relaxed dominance has been tested on different multiobjective optimisers and using a benchmark problem in order to

compare it to the conventional Pareto dominance. This issue is address here in this chapter.

The initial experimental results presented here show that *volume dominance* could work well on different EMO algorithms such as SEAMO2 [96], SPEA2 [124] and NSGA2 [47] when solving the multiple knapsack problem [125], a well-known benchmark multiobjective combinatorial optimisation problem. The details of these algorithms are given in chapter 3. In the experiments, the conventional Pareto dominance is replaced by the *volume dominance* to analyse the impact on the performance of these three algorithms. The experiments are aimed to investigate the performance of the *volume dominance* within the three evolutionary approaches with minimum alteration to the original algorithms. The replacement of the conventional Pareto dominance with the *volume dominance* in each algorithm is described below.

In SEAMO2, the *volume dominance* is used instead of Pareto dominance to decide on the replacement of a parent or a random solution by an offspring. This is the only stage where solutions are compared for dominance relationship in this algorithm. However, this is not the case for SPEA2 and NSGA2. In both SPEA2 and NSGA2, the fitness computation uses Pareto dominance whereas the mating scheme and the environmetal selection are based on the fitness value. It means that Pareto dominance is deployed in all these three stages. However the fitness computation using Pareto dominance is an integral part of these two algorithms, SPEA2 and NSGA2. Therefore, if *volume dominance* is used instead of Pareto dominance in the fitness computation it could lead to different EMO. It is not desired because this chapter is interseted in understanding if *volume dominance* could improve the performance of these EMO. The use of *volume dominance* concept to propose a new EMO is discussed in chapter 6. The environmental selection strategies in SPEA2 and NSGA2 employ additional techniques which relate to the density and the distance between solutions in the objective space. The mating

selection only use Pareto dominance (fitness value) to select parent. Therefore, it is beleived that applying *volume dominance* to this stage is the most appropriate in term of the degree of alteration to the original EMO algorithms. Therefore, in this preliminary investigation, the *volume dominance* is applied to the mating selection stage in SPEA2 and NSGA2. In other words, the comparison of individual fitnesses is conducted using the proposed *volume dominance* in order to decide on the superiority between individuals during the mating selection stage.

The 750 items and 4 objectives knapsack instance of the multiple 0/1 knapsack problem proposed in [125] is used. Short and long runs with medium and large population sizes and using different values of rSV is examined. Each short run uses 175,000 fitness evaluations and each long run uses 672,000 fitness evaluations. The values used for population size are 250 and 350 individuals. Six different values of rSV, 0.05, 0.10, 0.15, 0.20, 0.25 and 0.30 for the *volume dominance* are investigated. Regarding the values of rSV, other values (0.025, 0.075, 0.125, ...) are also examined but the results are very similar to those values addressed here in the thesis. Other values for rSV, which are greater than 0.30, are also examined but the results are extremely similar to the results obtained by Pareto dominance. The reason behind this is that when rSV is too big, the condition $r_{\vec{x},\vec{x}^*} > rSV$ is no longer valid, hence volume dominance becomes Pareto dominance. The reference point for *volume dominance* is chosen as the origin in the objective space. The results from 50 independent short runs and 30 independent long runs are summarised and discussed below.

The metrics for evaluating the nondominated fronts produced by the *volume dominance* and the conventional Pareto dominance approaches are *size of the space covered* $\mathcal{S}$ and *coverage of two sets* $\mathcal{C}$ [125]. The $\mathcal{S}$-metric, which measures the overall size of objective space covered by all nondominated solutions, is scaled as the percentage of the volume created by the origin and the reference point (41656, 40363, 41905, 41744) which is the profit sum of all items in each knapsack.

5.3.2 Results and Discussion

The boxplots in Figure 5.2, 5.4, 5.5 present the distribution of the reciprocal of the hypervolume $\mathcal{S}$-metric. Therefore, the vertical axes of the boxplots measure the nondominated objective space. The lower the boxplot, the better performance of the algorithm is. The horizontal axes present Pareto dominance (PD) and *volume dominance* (VD) with different rSV ratios. Each graph label in Figure 5.2, 5.4, 5.5 indicates the population size - the number of fitness evaluations.The average values (and the standard deviations in brackets) for the *coverage* $\mathcal{C}$-metric, which compares the dominance of the Pareto front obtained by one optimisation technique to that obtained by another optimisation technique, are given in Table 5.1, 5.2, 5.3. In these tables, the column labels present the population size - the number of fitness evaluation and the row labels VDx refer to *volume dominance* using $rSV = x/100$.

With respect to the *coverage* $\mathcal{C}$-metric, the performance of SEAMO2 with the proposed *volume dominance* and the Pareto dominance is quite similar as shown in Table 5.1. There is not any statistically significant difference between Pareto dominance and volume dominance regarding the *coverage* $\mathcal{C}$-metric. However, with respect to the hypervolume metric $\mathcal{S}$, Figure 5.2 shows that *volume dominance* in SEAMO2 suffers from using lower ratios rSV especially in longer runs. Using higher ratios rSV, *volume dominance* obtains competitive results compared to the conventional Pareto dominance, in both $\mathcal{S}$ and $\mathcal{C}$ metrics.

Figure 5.3 presents the distribution of each objective in the objective space for one particular run using SEAMO2 with population size of 250 in a long run. The vertical axis represents the objective value while the horizontal axis represents each objective with different types of dominance. For example, in the horizontal axis, pd1 stands for objective 1 of Pareto dominance, and vd5_1 stands for objective 1 of *volume dominance* using $rSV = 0.05$. A closer look at the values of individual objectives in the final nondominated sets, suggests that SEAMO2 using

the conventional Pareto dominance already obtains good extreme points in the objective space. Figure 5.3 shows that the sets of extreme points obtained with *volume dominance* using high ratios rSV are quite similar to the ones achieved with Pareto dominance. It seems that lower ratios rSV are less able to produce extreme points in this algorithm. However, *volume dominance* with low ratios rSV is better in pushing the set of the final nondominated solutions towards the Pareto front. In other words, the range of objective values is much better when using *volume dominance* with lower ratios rSV in SEAMO2. It indicates as smaller but higher (vertically) boxplots for *volume dominance* with lower ratios rSV.

It is predicted that the search strategy in SEAMO2, which outperforms two well-known algorithms SPEA2 and NSGA2 [96], should be able to obtain good extreme values in the objective space but its trade-off front shows a lot of variation. Figure 5.3 provides evidence to support this. It can be seen that there is a large number of outlier values in the boxplots for each objective when using Pareto dominance (PD). *Volume dominance* using low ratios rSV is less able to find outlier values. However, it is able to obtain a smoother trade-off front with considerably less variation. Figure 5.3 shows that the boxplots for *volume dominance* using ratios rSV of 0.05, 0.10 and 0.15 (VD5, VD10 and VD15 respectively) are much smaller than the ones for Pareto dominance in all 4 objectives. *Volume dominance* using higher ratios obtains similar distribution of objective values as those achieved by the conventional Pareto dominance.

In general, for SEAMO2 *volume dominance* is not capable of finding good extreme objective values as it is the case for Pareto dominance but it obtains a smoother trade-off front, especially when using lower ratios rSV. Furthermore, *volume dominance* helps SEAMO2 to push the trade-off front forward as a whole without bias on a particular objective. It is shown as smaller but higher boxplots for *volume dominance* with lower ratios rSV comparing these for Pareto dominance in figure 5.3.

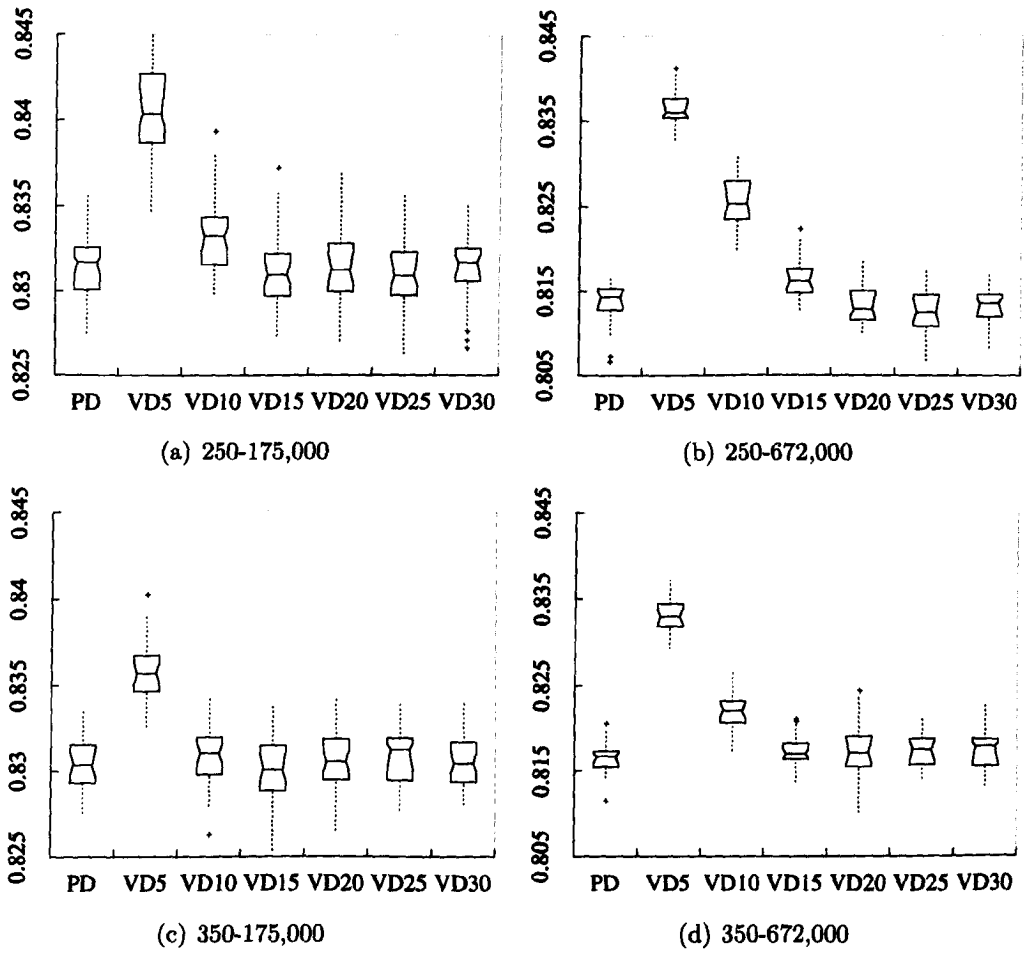


Figure 5.2: The performance of Pareto dominance and volume dominance on SEAMO2 for 4-objective knapsack problem on the reciprocal of the $\mathcal{S}$-metric.

Table 5.1: The performance of Pareto dominance and volume dominance on SEAMO2 for 4-objective knapsack problem on the $\mathcal{C}(A \succ B)$ metric.

SEAMO2 $\mathcal{C}(A \succ B)$					
Dominance		Population size - The number of fitness evaluations			
A	B	250-175,000	350-175,000	250-672,000	350-672,000
PD	VD5	20.6(34.3)	23.8(27.4)	1.7(4.8)	15.5(23.8)
	VD10	21.7(24.5)	21.9(25.0)	8.0(15.7)	18.4(22.6)
	VD15	25.6(33.7)	20.5(28.5)	11.9(14.8)	16.5(20.1)
	VD20	14.9(23.0)	25.4(25.9)	12.1(14.1)	19.7(23.4)
	VD25	18.4(26.3)	27.4(27.8)	17.2(17.3)	19.8(24.9)
	VD30	17.8(21.6)	26.7(27.7)	15.1(17.4)	14.8(18.6)
VD5	PD	18.0(26.9)	23.1(28.6)	20.0(17.8)	16.3(16.4)
VD10		17.0(24.7)	22.9(29.2)	17.6(16.8)	14.2(19.6)
VD15		23.4(30.3)	21.9(25.6)	18.7(19.6)	13.9(20.2)
VD20		23.6(22.8)	17.2(25.1)	14.1(17.2)	14.2(17.2)
VD25		23.3(26.7)	15.9(22.0)	11.8(13.4)	10.1(13.0)
VD30		20.3(24.9)	16.4(23.5)	16.4(19.3)	21.0(21.8)

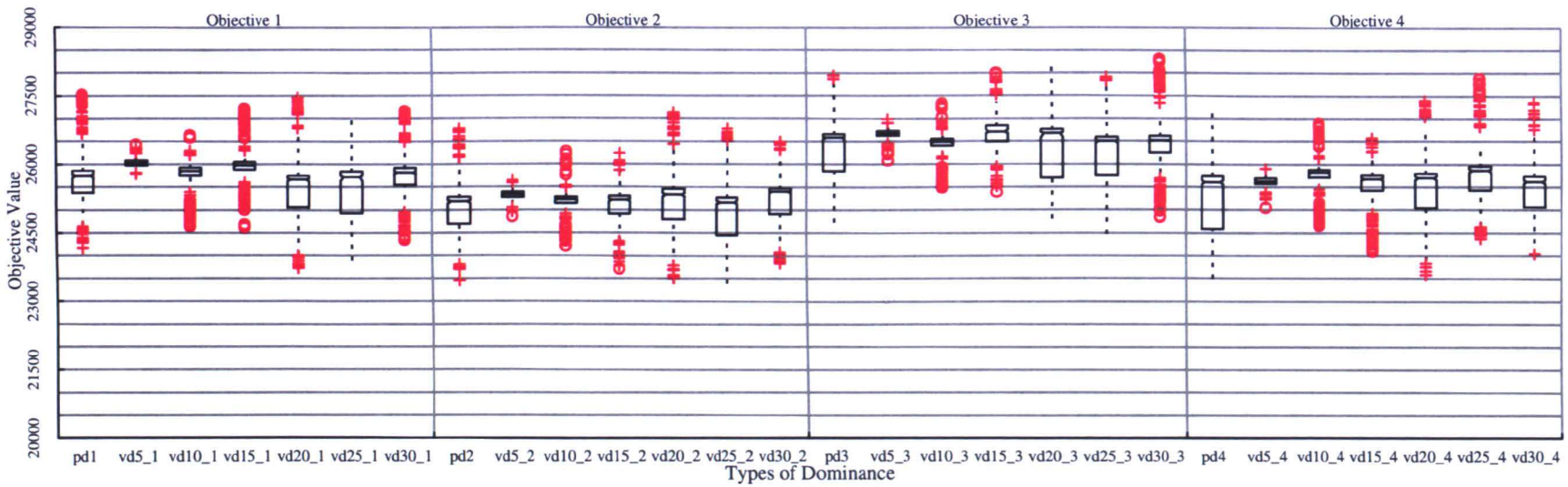


Figure 5.3: The distribution of objective values obtained when using conventional Pareto dominance and volume dominance on SEAMO2 for 4- objective knapsack problem.

Volume dominance deployed in SPEA2 and NSGA2 using low ratios rSV clearly outperforms Pareto dominance with respect to the C metric as seen in Tables 5.2 and 5.3. For example, in Table 5.2, for the run with population size of 250 and using 175,000 fitness evaluations (250-175,000 column), none of the nondominated solutions produced using Pareto dominance dominates solutions produced using *volume dominance* for the case when $rSV = 0.05$ (PD VD5 0(0)). On the other hand, based on the bottom 6 rows in the table, it can be seen that 78.1% of the nondominated solutions produced using *volume dominance* for the case when $rSV = 0.05$ dominate solutions produced using Pareto dominance with a standard deviation of 7.4% based on 50 independent runs (VD5 PD 78.1(7.4)). However, as in SEAMO2, the hypervolume of the final nondominated set obtained by deploying *volume dominance* with low ratios rSV in SPEA2 and NSGA2 is worse than the one obtained by deploying Pareto dominance, but not as bad as in SEAMO2 (Figures 5.4 and 5.5).

As in SEAMO2, the distribution of each objective value in the objective space when using SPEA2 and NSGA2 with the two types of dominance are presented in Figures 5.6 and 5.7 for a particular run using SPEA2 and NSGA2 respectively. As it is shown, *volume dominance* using low ratios rSV, deployed in SPEA2 and NSGA2, is slightly worse in obtaining extreme objective values than Pareto dominance. However *volume dominance* is better in pushing the trade-off front forward in all objectives and obtaining a smoother trade-off front. Figure 5.6 and 5.7 show a better range and a smaller size of the boxplot for *volume dominance* using $rSV = 0.05$ than for Pareto dominance.

Based on the results with respect to the *size of the space covered* $\mathcal{S}$, *coverage of two sets* C and the distribution of the trade-off front, it is suggested that the rSV ratio should be in the range of 0.15 to 0.20 for SEAMO2 and around 0.10 for SPEA2 and NSGA2.

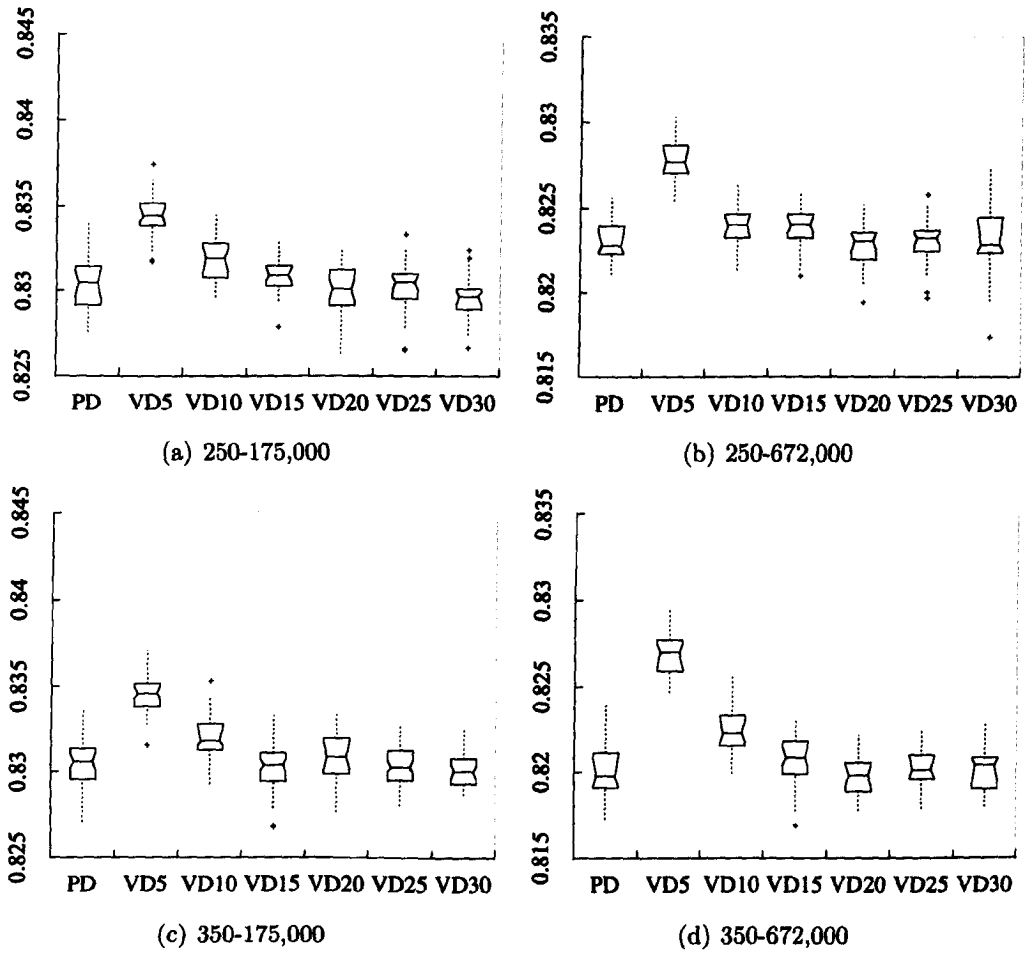


Figure 5.4: The performance of Pareto dominance and volume dominance on SPEA2 for 4-objective knapsack problem on the reciprocal of the $\mathcal{S}$-metric.

Table 5.2: The performance of Pareto dominance and volume dominance on SPEA2 for 4-objective knapsack problem on the $\mathcal{C}(\mathrm{A} \succ \mathrm{B})$ metric.

SPEA2 $\mathcal{C}(\mathrm{A} \succ \mathrm{B})$					
Dominance		Population size - The number of fitness evaluations			
A	B	250-175,000	350-175,000	250-672,000	350-672,000
PD	VD5	0(0)	0(0)	0(0)	0(0)
	VD10	4.8(5.8)	10.4(15.4)	0.9(1.3)	1.7(1.9)
	VD15	20.3(15.2)	23.2(18.5)	10.5(8.1)	14.9(7.2)
	VD20	19.6(14.3)	26.6(20.2)	20.2(11.8)	21.7(10.7)
	VD25	20.7(14.2)	24.7(16.3)	20.8(10.0)	22.8(12.5)
	VD30	24.9(23.7)	24.5(18.8)	24.0(11.3)	28.2(17.4)
VD5	PD	78.1(7.4)	77.3(6.6)	81.1(5.9)	73.7(8.3)
VD10		52.8(18.4)	48.4(22.8)	68.8(9.4)	61.8(12.7)
VD15		30.9(21.4)	31.4(22.2)	39.6(13.7)	33.6(11.6)
VD20		30.5(17.0)	27.7(20.1)	29.9(13.7)	26.2(11.1)
VD25		28.9(16.3)	27.6(16.9)	27.2(12.1)	26.3(11.5)
VD30		35.0(23.8)	30.8(19.7)	24.3(12.4)	26.8(18.0)

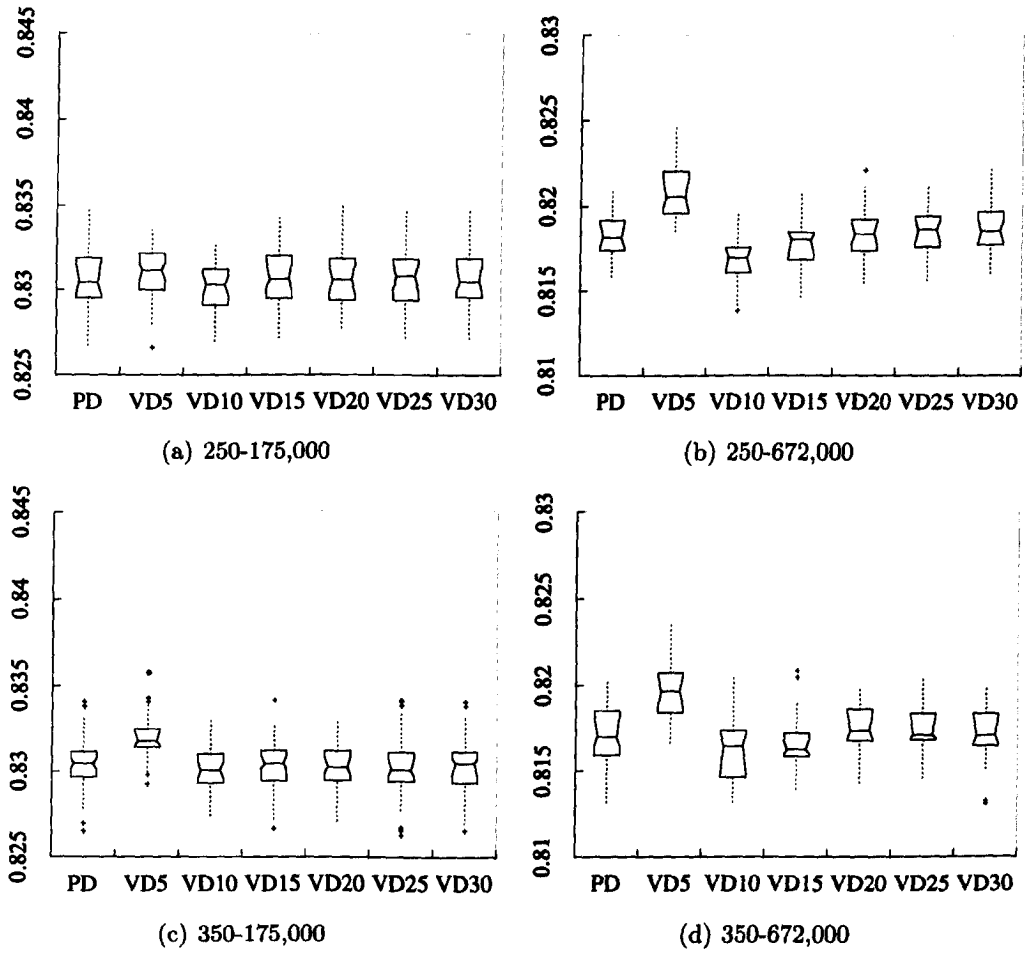


Figure 5.5: The performance of Pareto dominance and volume dominance on NSGA2 for 4-objective knapsack problem on the reciprocal of the $\mathcal{S}$-metric.

Table 5.3: The performance of Pareto dominance and volume dominance on NSGA2 for 4-objective knapsack problem on the $\mathcal{C}(A \succ B)$ metric.

NSGA2 $\mathcal{C}(A \succ B)$					
Dominance		Population size - The number of fitness evaluations			
A	B	250-175,000	350-175,000	250-672,000	350-672,000
PD	VD5	0(0)	0(0)	0(0)	0(0)
	VD10	0.9(2.1)	0.9(1.7)	0.1(0.2)	0.3(1.0)
	VD15	8.9(11.8)	13.5(10.9)	5.1(6.0)	4.9(6.5)
	VD20	17.3(13.5)	15.3(18.4)	9.3(10.1)	10.5(10.7)
	VD25	20.7(23.3)	49.4(40.6)	14.1(10.1)	14.2(8.7)
	VD30	78.1(37.3)	94.0(22.8)	14.3(8.0)	46.1(39.5)
VD5	PD	83.1(7.7)	84.0(7.2)	82.3(5.2)	78.7(12.6)
VD10		51.6(21.0)	51.6(16.6)	60.2(17.4)	53.5(19.1)
VD15		25.6(17.9)	23.7(16.7)	27.9(14.2)	29.1(16.6)
VD20		16.2(15.5)	22.1(17.4)	20.3(13.0)	22.9(16.9)
VD25		24.7(24.1)	47.4(42.4)	18.8(9.1)	15.6(11.0)
VD30		77.8(37.0)	95.4(17.6)	17.6(7.8)	43.7(41.1)

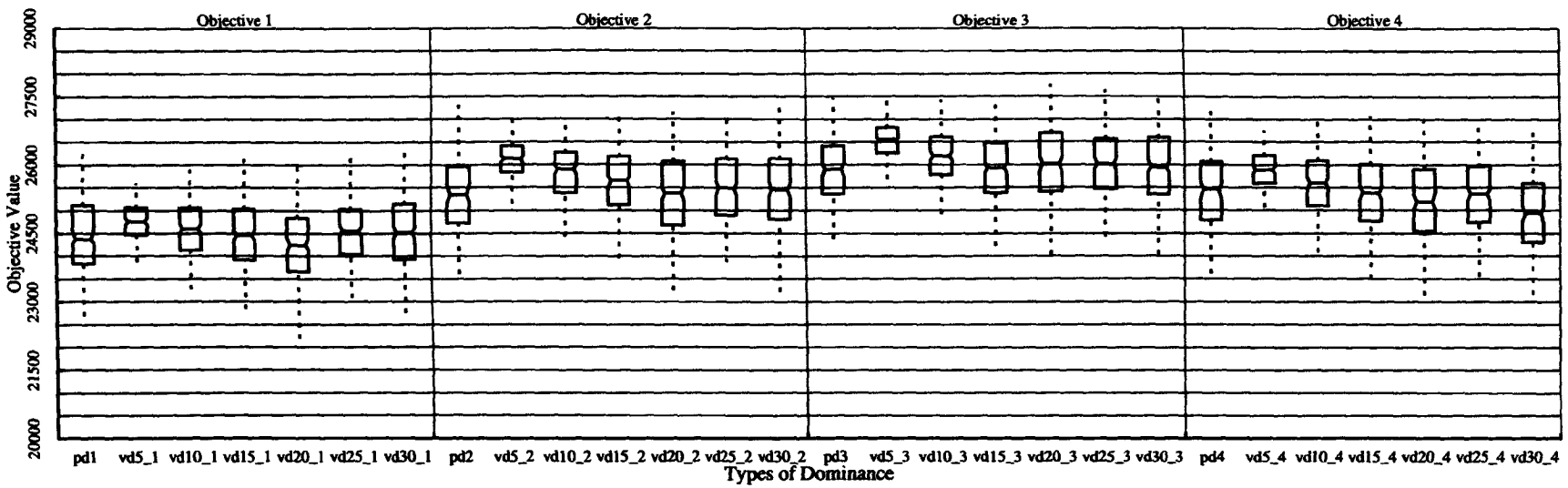


Figure 5.6: The distribution of objective values obtained when using conventional Pareto dominance and volume dominance on SPEA2 for 4-objective knapsack problem

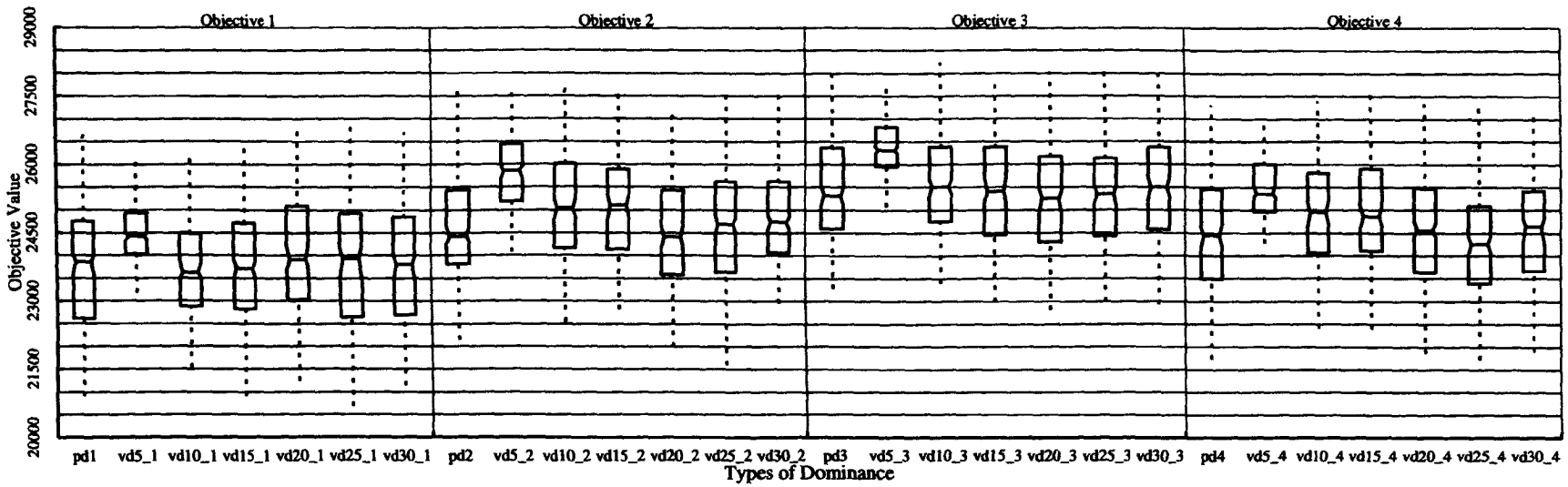


Figure 5.7: The distribution of objective values obtained when using conventional Pareto dominance and volume dominance on NSGA2 for 4-objective knapsack problem

5.3.3 Summary

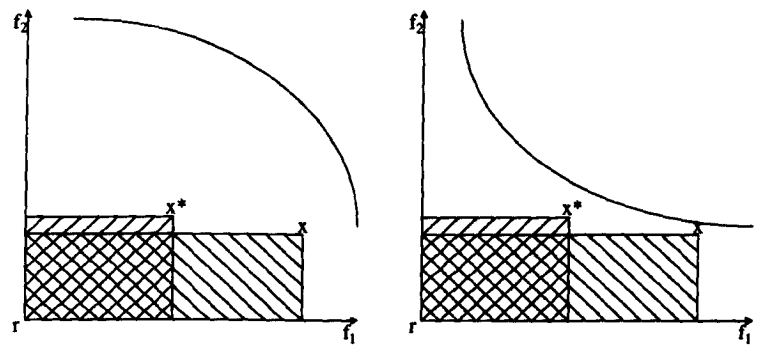
The initial results show that for the 4-objective knapsack problem with 750 items, the performance of *volume dominance* using high ratios rSV is quite similar to that of Pareto dominance. However, promising results are obtained by using lower ratios rSV. That is, *volume dominance* using low ratios rSV is capable of obtaining smoother trade-off fronts and is also able to 'push' the trade-off front in a more uniform manner than when using the conventional Pareto dominance, i.e. the trade-off front converges as a whole without bias on a particular objective. The results also suggest that *volume dominance* can be regarded as more robust than Pareto dominance because it helps the three algorithms implemented here to show more consistent performance compared to Pareto dominance which is indicated as smaller variations (smaller boxplots) for volume dominance using lower values for rSV in figures 5.3, 5.6 and 5.7. By adjusting the rSV ratio, users could obtain results driven by different criteria, such as a better coverage, a better size of space covered or a better distribution of the objective values.

5.4 Improved Volume Dominance

The proposed *volume dominance* presented in the previous section shows promising results when compared to the conventional Pareto dominance. It is shown that *volume dominance* is able to obtain results driven by different criteria such as better coverage, better size of space covered or better distribution of the objective values. This can be done by adjusting the rSV ratio. However, there are some drawbacks in the initial proposal, these weaknesses are discussed below.

First of all, *volume dominance* requires a preset reference point $\vec{r}$ in order to calculate the dominated volume of a solution. As the search progresses, the population moves away from the reference point and this can lead to a considerable

increase in the dominated volume of a solution. Hence, *volume dominance* could be highly effective at the start of the search but exert less and less influence as the search progresses and the current non-dominated front moves away from the reference point. Therefore, it is argued that the reference point should be adaptive to reflect the status of the current population. In other words instead of being fixed, the reference point should be defined based on some characteristics of the current population. Secondly, the initial proposal for *volume dominance* does not take into account the current shape Pareto front during the search process. This issue is illustrated in Figure 5.8.



(a) Concave (b) Convex

Figure 5.8: *Volume dominance.* In the case of the convex front, both points should be regarded as equally good instead of X being evaluated as better than X*

It is clear that for both cases 5.8(a) and 5.8(b) in Figure 5.8, $\vec{x} \succeq_V \vec{x^*}$ for some ratio rSV by using the initial approach. However, one can easily point out that $\vec{x} \succeq_V \vec{x^*}$ should not be the case in 5.8(b). As both $\vec{x}$ and $\vec{x^*}$ in 5.8(b) seem equally good (close to the Pareto front), which should be regarded as non-volume-dominated solutions. In order to overcome this issue, it is suggested that *volume dominance* should consider the current Pareto front (the set of nondominated solutions of the current population) while establishing superiority between two solutions. These issues are addressed below and a clustering technique is proposed as an additional feature incorporated into the *improved volume dominance.*

5.4.1 Improved Volume Dominance

5.4.1.1 Reference Point

Volume dominance proposed in section 5.3 requires a reference point $\vec{r}$ to calculate the dominated volume of a solution $\vec{x}$. The simple strategy is to define the reference point $\vec{r}$ as a fixed point, the origin of coordinates in the solution space. This simple strategy has the drawback of degrading the effectiveness of *volume dominance* as the search progresses. As the search progresses, the current population move towards the Pareto-optimal front and move away from the fixed reference point. It leads to an increase in the value of the shared dominated volume ($SV(\vec{x}, \vec{x^*})$) but not the difference between dominated volume of any pair of solutions ($V_{\vec{x}} - V_{\vec{x^*}}$). As a result, the value of the ratio $r_{\vec{x},\vec{x^*}} = \frac{V_{\vec{x}} - V_{\vec{x^*}}}{SV_{\vec{x},\vec{x^*}}}$ reduces considerably which make it more difficult to satisfy the second condition of *volume dominance* ($r_{\vec{x},\vec{x^*}} > rSV$). Therefore, it degrades the effectiveness of *volume dominance.* A more elaborate, but yet more effective, strategy to estimate the reference point $\vec{r}$ is discussed in this section. As the initial proposal, a common reference point is used to calculate the dominated volume for every solution in the population. However, the strategy proposed here has a designated reference point for each solution in the population. This strategy also considers the current state of the population in determining the reference point.

$$r_i^x = f_i(\vec{x}) - (r_i^{\sup} - r_i^{\inf}) \tag{5.4}$$

$$r_i^{\inf} = \inf\{f_i(\vec{x^*}) \mid \vec{x^*} \in P\} \tag{5.5}$$

$$r_i^{\sup} = \sup\{f_i(\vec{x^*}) \mid \vec{x^*} \in P\} \tag{5.6}$$

The reference point $\vec{r^x} = (r_1^x, r_2^x, \ldots, r_m^x)$ for solution $\vec{x} \in P$, the current population, is given in equation (5.4). The estimation of $\vec{r^x} = (r_1^x, r_2^x, \ldots, r_m^x)$ is illustrated in Figure 5.9 where $r^{\inf}$ and $r^{\sup}$ define a hyperbox.

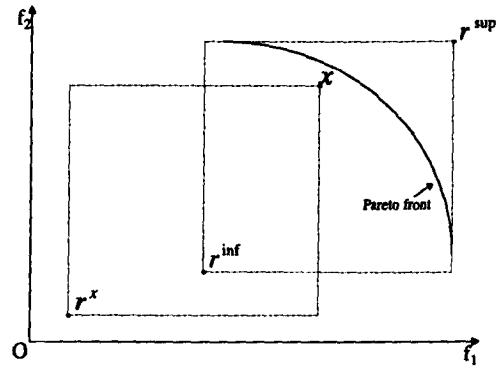


Figure 5.9: Volume Dominance - Reference Point.

5.4.1.2 Considering the Current Pareto Front

Without taking the current Pareto front into consideration during the search, Figure 5.8 illustrates the drawback of the initial proposed *volume dominance.* While establishing the superiority between solutions $\vec{x}$ and $\vec{x^*}$, the initial proposal defines the strength of solution $\vec{x}$ as the ratio of the dominated volume of $\vec{x}$ ($V(\vec{x})$) to the *shared dominated volume* of $\vec{x}$ and $\vec{x^*}$ ($SV(\vec{x}, \vec{x^*})$) with respect to the reference point $\vec{r}$. Then the dominance of $\vec{x}$ over $\vec{x^*}$ (or vice versa) is determined based on comparing the ratio of the difference between their strengths to rSV.

The *improved volume dominance* takes a different approach in defining the strength of solution $\vec{x}$. The strength of $\vec{x}$ is the ratio between the dominated volume of $\vec{x}$ ($V(\vec{x})$) and the volume that *fairly* represents the state of the current Pareto front. With respect to $\vec{x}$, this *fair* representation of the current Pareto front is the subset consisting of nondominated solutions that Pareto-dominate $\vec{x}$ (figure 5.10). However, determining the dominated volume of a solution set could be computationally expensive. Therefore, the dominated volume of this solution set is estimated as the dominated volume of the least solution x^{ref} that Pareto-dominates all solutions in that solution set.

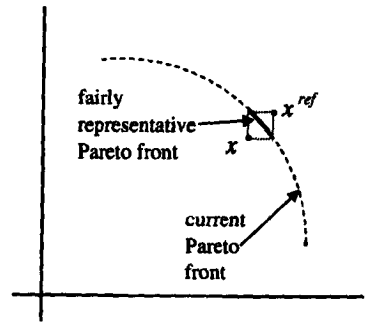


Figure 5.10: The fairly representative Pareto front of the population with respect to the individual $\vec{x}$.

Let us refer to this estimated dominated volume (w.r.t $\vec{x}$) as the reference volume of $\vec{x}$, $V^{ref}(\vec{x})$.

$$V^{ref}(\vec{x}) = \prod_{i=1}^{m}(x_i^{ref} - r_i^x) \tag{5.7}$$

$$x_i^{ref} = \inf\left\{\{f_i(\vec{x})\} \bigcup \{f_i(\vec{x^*}) \mid \vec{x^*} \succeq \vec{x} \wedge \vec{x^*} \in ParetoFront\}\right\} \tag{5.8}$$

The strength of $\vec{x}$ is then defined as follows:

$$Str(\vec{x}) = \frac{V(\vec{x})}{V^{ref}(\vec{x})} \tag{5.9}$$

Therefore $\vec{x}$ volume-dominates $\vec{x^*}$ ($\vec{x} \succeq_V \vec{x^*}$) if and only if the following condition holds for a positive ratio r_{Str}:

$$Str(\vec{x}) - Str(\vec{x^*}) \geq r_{Str} \tag{5.10}$$

To additionally ensure the improvement of the Pareto front, condition (5.10) is relaxed whenever $Str(\vec{x}) = 1$ and $Str(\vec{x^*}) < 1$. In other words, if none of nondominated solutions dominates $\vec{x}$, $x_i^{ref} = f_i(\vec{x})$ implying $V^{ref}(\vec{x}) = V(\vec{x})$, and it is not the case for $\vec{x^*}$ then $\vec{x} \succeq_V \vec{x^*}$.

It is noted that the current non-dominated front is required in order to apply this improved *volume dominance*. Therefore, one must use Pareto dominance to obtain the Pareto front and $\vec{x^{ref}}$ to estimate the reference volume of $\vec{x}$, $V^{ref}(\vec{x})$.

5.4.1.3 Clustering Strategy

This section proposes a clustering strategy as part of *volume dominance* that helps to improve the distribution of nondominated solutions w.r.t the objective space. This strategy is only considered when two solutions $\vec{x}$ and $\vec{x^*}$ are regarded as mutually non-volume-dominated. That is, for the case when both condition (5.10) and its relaxation do not hold for the pair of solutions $\vec{x}$ and $\vec{x^*}$. Solution $\vec{x}$ is said to dominate solution $\vec{x^*}$ if $\vec{x}$ is in a less crowded area comparing to $\vec{x^*}$. The degree of crowding of $\vec{x}$ is measured as the number of neighbours of $\vec{x}$. The number of neighbours of $\vec{x}$ is the number of nondominated solutions in the current non-dominated set that ϵ-dominates $\vec{x}$. The ϵ-dominance deployed here is slightly different from the one proposed by Laumanns et al. [82] and other variants of ϵ-dominance in the literature. The variants of ϵ-dominance either deploy a dynamic adaptation for ϵ value or a different ϵ_i value for each objective. The variant of ϵ-dominance employed here takes the advantage of both approaches, a different dynamic adaptive ϵ_i value for each objective. The ϵ_i value is estimated based on the current Pareto front as follows:

$$\epsilon_i = (r_i^{\text{sup}} - r_i^{\text{inf}}) \times \mu \tag{5.11}$$

where r_i^{inf}, r_i^{sup} given in (5.5), (5.6) respectively and μ is a positive constant. Within the context of *volume dominance*, it is said that $\vec{x}$ ϵ-dominates $\vec{x^*}$ ($\vec{x} \succeq_{\epsilon_v} \vec{x^*}$) if and only if $f_i(\vec{x}) \geq f_i(\vec{x^*}) - \epsilon_i$ $\forall i = 1, \ldots, m$ and $f_i(\vec{x}) > f_i(\vec{x^*}) - \epsilon_i$ for at least one $i = 1, \ldots, m$. Then the number of neighbours of $\vec{x}$ is defined as follows:

$$N(\vec{x}) = \left|\left\{\vec{x^*} \mid \vec{x^*} \succeq_{\epsilon_v} \vec{x} \ \wedge \ \vec{x^*} \in ParetoFront\right\}\right| \tag{5.12}$$

It is then said that if condition (5.10) and its relaxation do not hold for either the pair $(\vec{x}, \vec{x^*})$ or $(\vec{x^*}, \vec{x})$, then $\vec{x}$ volume-dominates $\vec{x^*}$ ($\vec{x} \succeq_V \vec{x^*}$) for a positive constant τ if and only if:

$$N(\vec{x^*}) - N(\vec{x}) \geq \tau \tag{5.13}$$

5.4.2 Experimental Design

The *improved volume dominance* is incorporated into 3 algorithms SEAMO2 [96], SPEA2 [124] and NSGA2 [47]. The 750 items and 2-, 3-, and 4-objective instances of the knapsack problem proposed by Zitzler and Thiele [125] are used. Then the *improved volume dominance* is investigated based on short and long runs using different values of rSV. The population size used for the 2-, 3-, and 4-objective instances are 250, 300 and 350 individuals respectively. The number of generations is 500 generations (short run) and 1920 generations (long run) as used by Zitzler et al. [124], Deb et al. [47] and Mumford [96]. For the *improved volume dominance*, 5 different values of $rSV = \{0.025, 0.05, 0.075, 0.10, 0.15\}$ in inequality (5.10), $\mu = 0.01$ in equation (5.11) and $\tau = 5$ in inequality (5.13) are used. For the initial proposed *volume dominance*, four different values of $rSV = \{0.10, 0.15, 0.20, 0.25\}$ are used. The results from 30 independent runs are summarised and discussed. The results in section 5.4.3 are based on $rSV = 0.075$ using the new approach for the *volume dominance* and $rSV = 0.15$ for the initial approach of *volume dominance*.

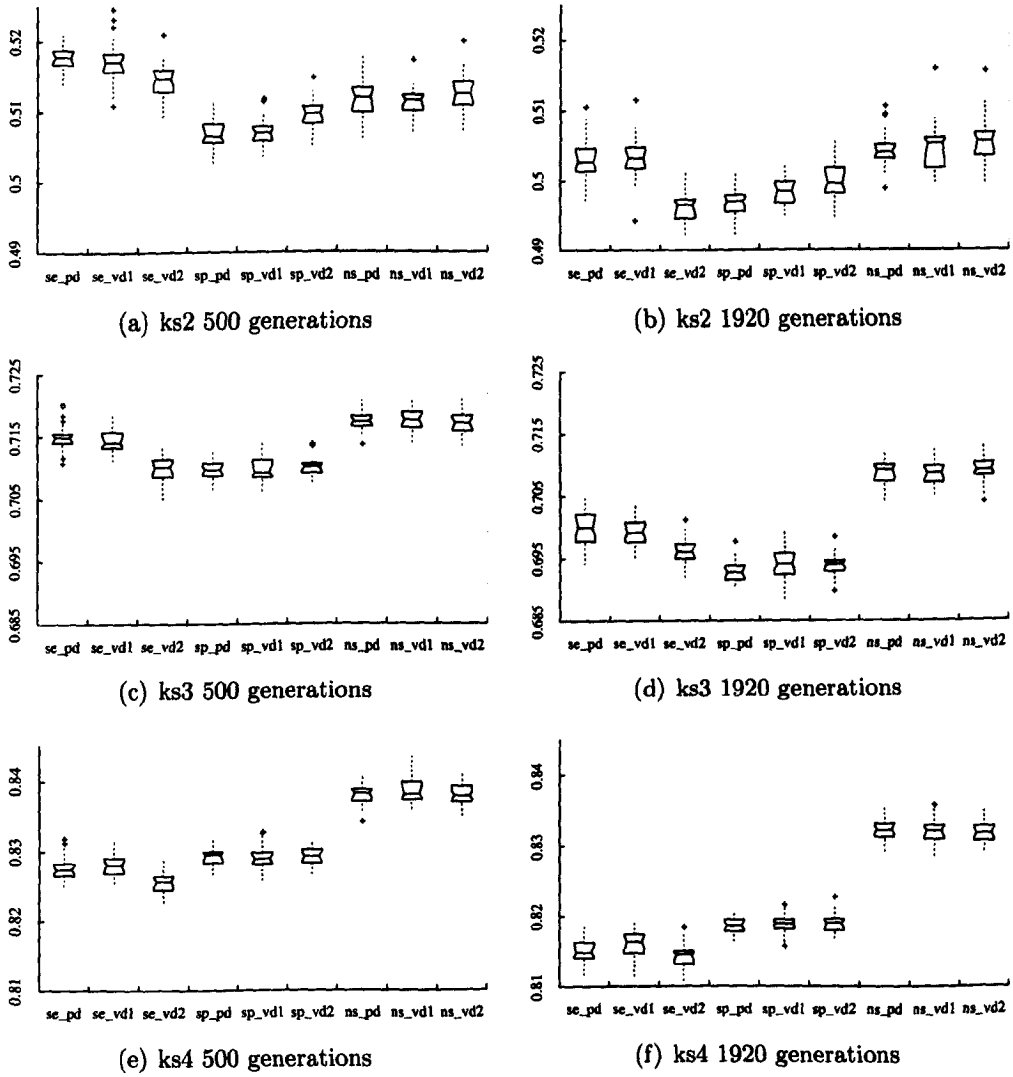
Four metrics are used to evaluate the nondominated fronts produced. The first metric is the $\mathcal{S}$ hypervolume proposed by Zitzler and Thiele [125] which measures the overall size of the objective space covered by all the nondominated solutions. Here, $\mathcal{S}$ is scaled as the percentage of the volume created by the origin and the reference point (39822, 41166), (41968, 41298, 41402), (41841, 40790, 39651, 41630) which is the sum profits of all items in each objective for 2-, 3- and 4-objective instance respectively. The second metric is the *cluster* CL_μ proposed by Wu and Azarm [119]. The CL_μ cluster metric indicates the average number of *indistinct* solutions in each small grid which size is specified by $1/\mu$. The ideal case is $CL_\mu = 1$ which means every obtained Pareto solution is distinct. In all other cases, CL_μ is greater than 1. The higher value of CL_μ is, the more clustered the solution set is, and therefore the less preferred the solution set. Here μ is

set to 0.01 or in other words $1/\mu = 100$ units in the objective space. The third metric is the average distance from the obtained nondominated front to the Pareto-optimal front. However, the Pareto-optimal front is not available for every instance. Therefore, as aforementioned in section 3.3.2, the approximation of the Pareto-optimal front could be used instead. The lower value of this metric is, the better convergence, that is, the closer the obtained nondominated front is to the true Pareto front. Finally, the size of the nondominated fronts is assessed.

5.4.3 Results and Discussion

The boxplots in Figure 5.11 represent the distribution of the reciprocal of the hypervolume $\mathcal{S}$-metric $(1 - \mathcal{S})$. The vertical axes of the boxplots measure the percentage of the nondominated objective space. The lower the boxplot, the better performance of the algorithm is. The horizontal axes present Pareto Dominance (pd), the previous *volume dominance* (vd1) and the *improved volume dominance* proposed here (vd2) applying to three different evolutionary algorithms SEAMO2 (se), SPEA2 (sp) and NSGA2 (ns).

Regarding the hypervolume $\mathcal{S}$-metric (Figure 5.11), the *improved volume dominance* incorporated into SEAMO2 (se_vd2) outperforms not only the Pareto dominance (se_pd) but also the previous *volume dominance* (se_vd1). The se_vd2 outperforms the se_pd and the se_vd1 for all knapsack instances both in the short and long runs. The *improved volume dominance* when incorporated into NSGA2 (ns_vd2) is slightly worst than ns_pd and ns_vd1 in 2-knapsack instance but ns_vd2 is able to compete against ns_pd and ns_vd1 in higher dimension knapsack instances (3- and 4-objective). It is observed that the same result in SPEA2 as in NSAG2 when comparing se_vd2 to se_pd and se_vd1.



(a) ks2 500 generations

(b) ks2 1920 generations

(c) ks3 500 generations

(d) ks3 1920 generations

(e) ks4 500 generations

(f) ks4 1920 generations

Figure 5.11: Performance of Pareto dominance, previous *volume dominance* and new improved *volume dominance* on SEAMO2, SPEA2, NSGA2 for 2-, 3-, and 4-objective knapsack problems with 750 items on the reciprocal of the $\mathcal{S}$-metric.

The performance of vd2 when incorporated in SPEA2 and NSGA2 is quite similar to vd1 and pd with respect to the size of the nondominated set (Table 5.4(b), 5.4(c)) as well as the cluster CL_μ of the nondominated set (Table 5.5(b), 5.5(c)). However, for SEAMO2, vd2 is noticeably better than vd1 and pd regarding both the size of the nondominated set (Table 5.4(a)) and the cluster CL_μ of the nondominated set (Table 5.5(a)) in almost all knapsack instances for both short and long runs, except for the 2-knapsack instance when vd2 is worse than vd1 and pd in term of the cluster CL_μ of the nondominated set.

Table 5.4: Average size (standard deviation) of the nondominated set.

(a) SEAMO2

ks	generations	se_pd	se_vd1	se_vd2
2	500	59.37 (8.1)	59.53 (8.44)	82.9 (9)
2	1920	101.67 (11.37)	104.5 (8.87)	198.9 (16.51)
3	500	199 (19.04)	202.73 (16.29)	299.87 (0.57)
3	1920	244.7 (10.54)	244.87 (11.11)	300 (0)
4	500	284.5 (15.87)	286.77 (15.79)	350 (0)
4	1920	321.27 (8.45)	316.2 (10.24)	350 (0)

(b) SPEA2

ks	generations	sp_pd	sp_vd1	sp_vd2
2	500	76.6 (8.81)	77.27 (8.28)	70.9 (8.94)
2	1920	134.63 (10.42)	131.9 (14.22)	126.1 (14.35)
3	500	300 (0)	300 (0)	300 (0)
3	1920	300 (0)	300 (0)	300 (0)
4	500	350 (0)	350 (0)	350 (0)
4	1920	350 (0)	350 (0)	350 (0)

(c) NSGA2

ks	generations	ns_pd	ns_vd1	ns_vd2
2	500	66 (8.29)	66.1 (8.19)	62.27 (7.73)
2	1920	85.83 (12.5)	73.2 (21.44)	73.63 (16.96)
3	500	257.37 (9.92)	255.7 (12.55)	256.47 (8.17)
3	1920	272.07 (7.52)	269 (5.38)	266.87 (6.43)
4	500	335.7 (5.64)	335.8 (6.72)	334.37 (5.88)
4	1920	338.7 (3.19)	338.9 (2.45)	338.07 (4.81)

Table 5.5: Average size (standard deviation) of the cluster metrics CL_{μ}.

(a) SEAMO2

ks	generations	se_pd	se_vd1	se_vd2
2	500	5.78 (0.79)	5.69 (0.7)	6.1 (0.8)
2	1920	6.16 (1.05)	6.49 (0.94)	8.74 (0.74)
3	500	7.23 (1.08)	7.25 (0.82)	3.95 (0.38)
3	1920	6.6 (0.71)	6.29 (0.6)	3.31 (0.23)
4	500	5.8 (0.66)	6.1 (0.63)	3.06 (0.31)
4	1920	5.2 (0.35)	5.29 (0.43)	2.77 (0.16)

(b) SPEA2

ks	generations	sp_pd	sp_vd1	sp_vd2
2	500	4.15 (0.45)	4.34 (0.52)	4.32 (0.54)
2	1920	6.17 (0.56)	6.16 (0.61)	6.34 (0.72)
3	500	2.09 (0.16)	2.14 (0.25)	2.16 (0.21)
3	1920	1.66 (0.09)	1.69 (0.08)	1.72 (0.09)
4	500	1.45 (0.06)	1.43 (0.06)	1.46 (0.06)
4	1920	1.36 (0.05)	1.38 (0.06)	1.35 (0.05)

(c) NSGA2

ks	generations	ns_pd	ns_vd1	ns_vd2
2	500	4.52 (0.57)	4.52 (0.55)	4.48 (0.46)
2	1920	5.04 (0.69)	4.26 (1.02)	4.42 (0.89)
3	500	2.99 (0.25)	3 (0.3)	3.22 (0.36)
3	1920	1.6 (0.07)	1.58 (0.07)	1.6 (0.07)
4	500	1.97 (0.16)	1.9 (0.13)	1.99 (0.17)
4	1920	1.78 (0.14)	1.77 (0.13)	1.82 (0.12)

It is also pointed out here that se_vd2, comparing to se_vd1 and se_pd, is able to not only find more nondominated solutions but also reduce the clustering in the nondominated set. In other words, se_vd2 is able to obtain more diverse solution sets and better extreme solutions. This promising result is accredited to the clustering strategy deployed in the *improved volume dominance.*

It is understandable that the average distance of the nondominated set found by vd2 is higher than vd1 and pd incorporated into SEAMO2 (table 5.6(a). As it was aforementioned, it is argued here that se_vd2 is able to find more extreme solutions than se_vd1 and se_pd. Finding more extreme solutions could be interpreted as obtaining more solutions that are slightly away from the approximation of the Pareto fronts. This is the reason why se_vd2 is not competitive with se_vd1 and se_pd with respect to the average distance from the nondominated set to the approximation of the true Pareto front (table 5.6(a)). However, table 5.6(b), 5.6(c) show that vd2 clearly outperforms vd1 and pd, when incorporated into SPEA2 and NSGA2 in all knapsack instances for both short and long runs.

Table 5.6: Average distance (standard deviation) from the nondominated set to the approximation of the true Pareto Front.

(a) SEAMO2

ks	generations	se_pd	se_vd1	se_vd2
2	500	498.88 (51.62)	491.97 (40.5)	602.42 (70.93)
2	1920	389.18 (35.34)	387.93 (34.69)	432.57 (43.93)
3	500	1381.9 (58.15)	1392.34 (68.96)	1506.43 (72.99)
3	1920	1250.8 (41.71)	1269.06 (35.85)	1318.78 (50.35)
4	500	744.97 (51.94)	733.9 (50)	808.94 (48.64)
4	1920	677.02 (55.06)	668.83 (65.52)	695.78 (33.22)

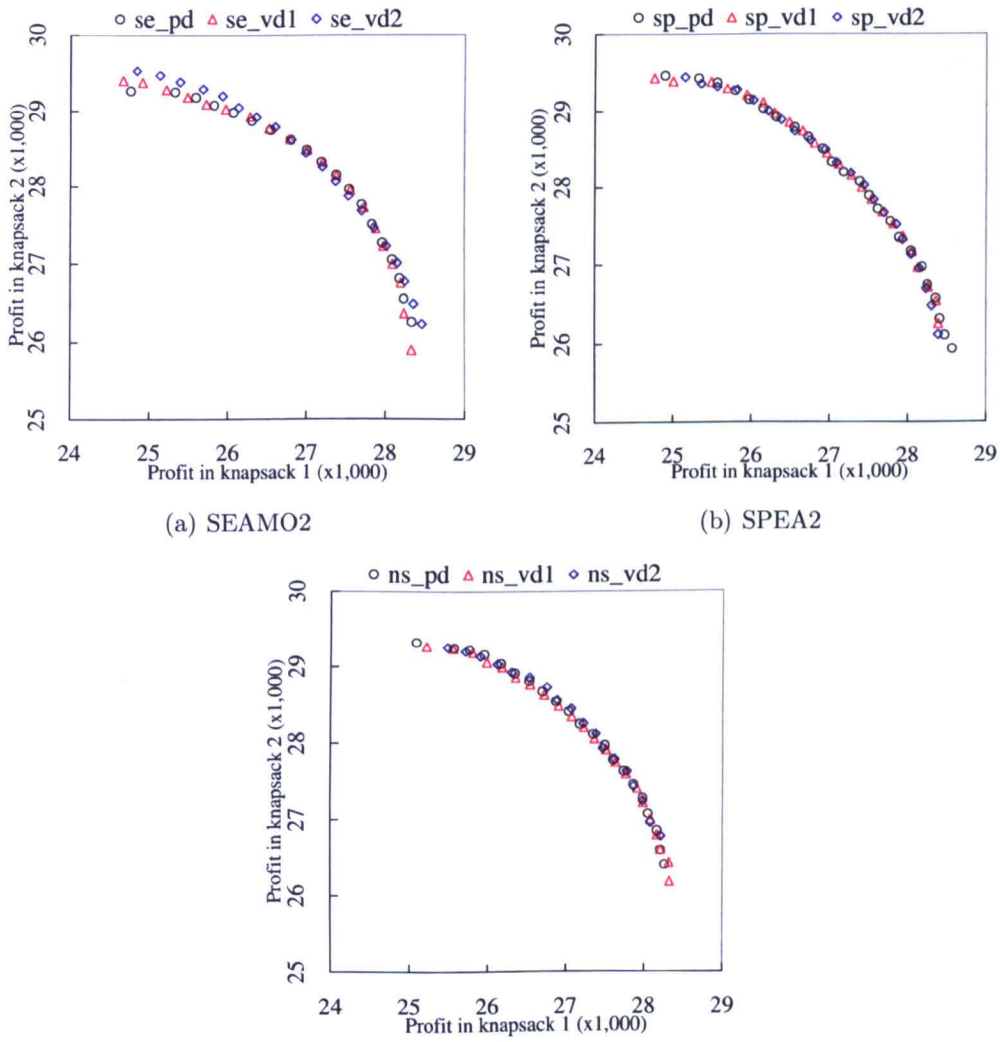
(b) SPEA2

ks	generations	sp_pd	sp_vd1	sp_vd2
2	500	701.4 (53.14)	676.4 (54.15)	638.26 (50.76)
2	1920	466.5 (43.39)	468.89 (53.54)	450.24 (43.8)
3	500	1985.72 (86.65)	1984.3 (68.79)	1960.08 (80.6)
3	1920	1682.92 (47.05)	1687.89 (34.42)	1692.77 (42.44)
4	500	1825.8 (100.38)	1761.69 (98.71)	1769.54 (115)
4	1920	1605.61 (55.04)	1567.62 (67.82)	1571.16 (70.83)

(c) NSGA2

ks	generations	ns_pd	ns_vd1	ns_vd2
2	500	613.85 (39.39)	623.58 (43.67)	592.18 (54.07)
2	1920	429 (42.49)	454.39 (39.32)	429.03 (50.83)
3	500	2029.53 (80.73)	2026.49 (90.53)	1933.72 (99.73)
3	1920	1739.44 (74.74)	1729.7 (77.4)	1687.68 (69.35)
4	500	1681.5 (135.55)	1711.23 (143.16)	1640.75 (110.29)
4	1920	1316.23 (95.44)	1290.11 (89.45)	1285.49 (91.13)

Figure 5.12 shows the offline results for the 2-knapsack instance. They are the combined nondominated solutions from 30 long runs . For better visualisation, we show the nondominated fronts in a lower density. See that vd2, vd1 and pd are quite similar when incorporated into SPEA2 and NSGA2 (Figure 5.12(b), 5.12(c)) but Figure 5.12(a) shows a better performance of vd2 over vd1 and pd.



(a) SEAMO2

(b) SPEA2

(c) NSGA2

Figure 5.12: The combined non-dominated front obtained from 30 independent runs for 2-objective knapsack problem with 750 items.

5.5 Summary

This chapter proposed a new form of *relaxed Pareto dominance*, called *volume dominance.* Extensive experiments are presented to compare the performance of both the initial proposal and the improved proposal of the *volume dominance* approach, an alternative to the conventional Pareto dominance, using three EMO algorithms, SEAMO2, SPEA2 and NSGA2. The results show that in most of the cases, using 3 different knapsack instances using short and long runs, the new *improved volume dominance* approach performs better than the Pareto dominance and the initial proposed *volume dominance.* The *improved volume dominance* is more effective when incorporated into SEAMO2 than when incorporated into SPEA2 and NSGA2. This could be due to the fact that SEAMO2 is a very simple strategy whereas SPEA2 and NSGA2 already deploy more elaborate mechanisms. It is suggested that this *improved volume dominance* could be used as a new strategy to assign fitness to solutions in multiobjective evolutionary algorithms. The next chapter will discuss in more details and propose a new EMO algorithm.

Chapter 6

Hyper Volume Evolutionary Algorithm

The previous chapter proposes *volume dominance*, a new form of relaxed Pareto dominance. It also suggests that the strategies associated to the *improved volume dominance* could be used to compute solutions' fitness and crowding. This chapter further explores these ideas and proposes a new EMO algorithm, named the Hyper Volume Evolutionary Algorithm (HVEA). HVEA is a population-based EMO algorithm. The algorithm is characterised by:

- An individual's fitness evaluation dependent on the current Pareto front of the population.
- A ranking strategy to classify individuals based on their fitness.
- A crowding assignment for individuals to preserve population diversity.

The fitness of an individual indicates how close that individual is to the current Pareto front based on the ratio of its dominated hypervolume to the one of the current representative Pareto front of the population. The current representative Pareto front of the population with respect to an individual is the part the current

Pareto front contained nondominated individuals which dominate that individual (figure 6.1). The population is then ranked based on individual fitnesses rather than non-dominated sorting of the population. The crowding assignment takes into account the distances between an individual to all other solutions in its neighbourhood which is defined by a parameter ω, the *neighbouring area radius*. Extensive experiments on the multiple 0/1 knapsack problem using different greedy repair methods (as described in section 3.3.1) are presented to compare the performance of HVEA to other state-of-the-art EMO algorithms including NSGA2, SEAMO2, SPEA2, IBEA and MOEA/D.

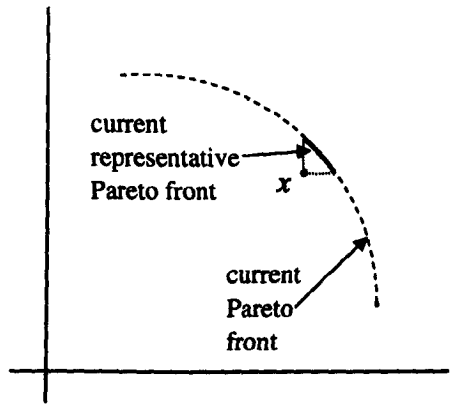


Figure 6.1: The current representative Pareto front of the population with respect to the individual $\vec{x}$

6.1 Hyper Volume Evolutionary Algorithm

The *Hyper Volume Evolutionary Algorithm* (HVEA) is a new approach to multiobjective optimisation. HVEA exploits the concept of the *improved volume dominance* (here, *volume dominance* for short) proposed in the previous chapter (chapter 5). HVEA deploys techniques that are well established in the literature as well as presents new mechanisms in order to find the Pareto front of the multiobjective problem in hand. As other population-based EMO algorithms, HVEA

- uses a population for the generated offspring and an archive for keeping the best solutions found so far.
- assigns scalar fitness values to individuals based on the Pareto dominance relationship.
- employs a crowding strategy to maintain the diversity of the population and, if necessary, maintain the size of the archive during the truncation procedure.

However, HVEA is distinguished by four characteristics:

- The fitness of an individual is determined by the hyper volume of that individual and the hyper volume of the current representative Pareto front.
- Individuals are ranked based on their fitness value. Individuals having fractional differences in fitness values are classified into the same rank.
- A new niching technique to preserve the diversity of the population is distance based. This technique takes into account distances from one particular individual to all other individuals in the population to determine the crowding of the area around that individual.
- Offspring, which improve the current Pareto front (move the Pareto front forward, or in other words, offspring that dominate individuals in the Pareto front), are guaranteed to be in the archive during the environmental selection and to be chosen during the tournament selection to fill the mating pool.

The main loop of HVEA is as follows:

Step 1: *Initialisation:* Generate an empty offspring population $\boldsymbol{P}$ and create an initial archive $\overline{\boldsymbol{P}}$ consisting of random generated individuals (each of size N).

Step 2: *Fitness Assignment:* Calculate fitness values of individuals in $\boldsymbol{P} \bigcup \overline{\boldsymbol{P}}$ (*cf. Section 6.1.1*).

Step 3: *Ranking Assignment:* Assign rank to each individual in $\boldsymbol{P} \bigcup \overline{\boldsymbol{P}}$ (*cf. Section 6.1.2*).

Step 4: *Environmental Selection:* Repeatedly copy all individuals having the best rank to $\overline{\boldsymbol{P}}$ and assign their crowding values until the size of $\overline{\boldsymbol{P}}$ exceeds N. Then reduce the size of $\overline{\boldsymbol{P}}$ by means of the truncation operator (*cf. Section 6.1.3* and *Section 6.1.4*).

Step 5: *Termination:* If the stopping criteria are satisfied then present all non-dominated individuals in $\overline{\boldsymbol{P}}$ as solutions to the problem and stop the algorithm. Otherwise, go to **Step 6**.

Step 6: *Offspring Generation:* Apply crossover and mutation on parents, which are chosen by binary tournament selection, to produce offspring (*cf. Section 6.1.5*). Go to **Step 2**.

6.1.1 Fitness Assignment

The fitness assignment procedure in HVEA deploys the concept of *volume dominance* proposed in the previous chapter. Based on the concept of *volume dominance*, each individual in the population is assigned a *strength* value which is defined as the ratio between the hyper volume of that individual and the hyper volume of the current representative Pareto front of the population. The dominance relationship between two individuals is established by comparing their respective *strengths* with respect to the threshold strength r_{Str}. In more detail, the fitness assignment of HVEA works as follows. Let:

ParetoFront is the Pareto front of the combining offspring population $\boldsymbol{P}$ and

archive $\overline{\boldsymbol{P}}$.

$\boldsymbol{RP}(\vec{x})$ is the current representative Pareto front, the set of all individuals in ***ParetoFront*** that Pareto-dominate $\vec{x}$.

$$\boldsymbol{RP}(\vec{x}) = \left\{\vec{x^*} \mid \vec{x^*} \succeq \vec{x} \bigwedge \vec{x^*} \in \boldsymbol{ParetoFront}\right\} \tag{6.1}$$

$V(\vec{x})$ is the hyper volume of individual $\vec{x}$ w.r.t. the reference point $\vec{r^x}$.

$V(\vec{x^{ref}})$ is the hyper volume of $\vec{x^{ref}}$, or the reference hyper volume of individual $\vec{x}$, w.r.t. the reference point $\vec{r^x}$.

$$x_i^{ref} = \inf\left\{\{f_i(\vec{x})\} \bigcup \{f_i(\vec{x^*}) \mid \vec{x^*} \in \boldsymbol{RP}(\vec{x})\}\right\} \tag{6.2}$$

Then, the fitness value of individual $\vec{x}$ is calculated as follows:

$$fitness(\vec{x}) = 1 - \frac{V(\vec{x})}{V(\vec{x^{ref}})} \tag{6.3}$$

As $\boldsymbol{RP}(\vec{x})$ is an empty set if $\vec{x}$ is a Pareto solution, equation (6.3) is relaxed by setting $fitness(\vec{x}) = 0$. The estimation of the reference point $\vec{r^x} = (r_1^{\vec{x}}, r_2^{\vec{x}}, \ldots, r_m^{\vec{x}})$ and the hyper volume $V(\vec{x})$, $V(\vec{x^{ref}})$ (as given in section 5.4.1.2 chapter 5) is as follows:

$$r_i^{\vec{x}} = f_i(\vec{x}) - (r_i^{\max} - r_i^{\min}) \tag{6.4}$$

$$r_i^{\min} = \min\left\{f_i(\vec{x^*}) \mid \vec{x^*} \in \boldsymbol{P} \bigcup \overline{\boldsymbol{P}}\right\} \tag{6.5}$$

$$r_i^{\max} = \max\left\{f_i(\vec{x^*}) \mid \vec{x^*} \in \boldsymbol{P} \bigcup \overline{\boldsymbol{P}}\right\} \tag{6.6}$$

$$V(\vec{x}) = \prod_{i=1}^{m}(f_i(\vec{x}) - r_i^{\vec{x}}) \tag{6.7}$$

$$V(\vec{x^{ref}}) = \prod_{i=1}^{m}(x_i^{ref} - r_i^{\vec{x}}) \tag{6.8}$$

The lower value of $fitness(\vec{x})$ is preferred. The 2-step procedure to determine the fitness values of individuals in the offspring population $\boldsymbol{P}$ and the archive $\overline{\boldsymbol{P}}$ is as follows:

Step 1: Use all individuals in $\boldsymbol{P}$ and $\overline{\boldsymbol{P}}$ to update:

$$\vec{r^{\min}} = \left(r_1^{\min}, r_2^{\min}, \ldots, r_m^{\min}\right) \tag{6.9}$$

$$\vec{r^{\max}} = \left(r_1^{\max}, r_2^{\max}, \ldots, r_m^{\max}\right) \tag{6.10}$$

where $r_i^{\min}$ and $r_i^{\max}$ are given by equation (6.5) and (6.6) respectively.

Step 2: Update ***ParetoFront*** using all newly generated offspring in $\boldsymbol{P}$. Assign fitness value of -1 to offspring that Pareto-dominate at least one individual in ***ParetoFront*** of the previous generation. Assign fitness value of 0 to the remaining individuals in ***ParetoFront*** of the current generation. Apply equation (6.3) to assign fitness value to all remaining individuals in $\boldsymbol{P}$ and $\overline{\boldsymbol{P}}$.

Assigning fitness value of -1 to offspring that Pareto-dominate at least one individual in ***ParetoFront*** enables HVEA to distinguish individuals that make a direct contribution towards moving the Pareto front forward during the environmental selection and mating selection. This fitness assignment strategy emphasises the idea of preferring individuals near the Pareto front, especially individuals that improve the Pareto front from the previous generation to the current one.

6.1.2 Ranking Assignment

As commonly found in the literature, the ranking assignment in EMO classifies individuals with similar (or identical) characteristic(s) into the same category(ies). The characteristic usually used is the Pareto non-domination. In other words,

Pareto non-dominated individuals are classified into one category (rank). Therefore, it is guaranteed that individuals with the same rank are all Pareto non-dominated. An example is NSGA2 [47] with its fast non-dominated sorting approach. HVEA takes a paradigm shift in ranking individuals. HVEA does not guarantee this property. In HVEA, there is a case that an individual is Pareto dominated by another individual having the same rank. The main intention of this mechanism is to allow slightly worse quality individuals to be able to compete for survival and/or selection. HVEA computes ranking of an individual directly from its fitness value as follows:

$$rank(\vec{x}) = \left\lfloor fitness(\vec{x}) \times \frac{1}{\mu} \right\rfloor \tag{6.11}$$

Parameter μ indicates the "size" of each ranking level, whereas $\frac{1}{\mu}$ could be referred as the desired number of ranks (desired number of "fronts"). The advantage of this ranking assignment mechanism is that only Pareto non-dominated individuals are assigned rank 0 with the exception of individuals, which improve the Pareto front, the ranking for these individuals is set to -1. Other Pareto dominated individuals are assigned rank with some noise induction by applying equation (6.11).

6.1.3 Environmental Selection

Let $\mathcal{F}_i$ be the set of individuals in $\boldsymbol{P} \bigcup \overline{\boldsymbol{P}}$ with $rank(\vec{x}) = i$

$$\mathcal{F}_i = \left\{ \vec{x} \mid rank(\vec{x}) = i \wedge \vec{x} \in \boldsymbol{P} \bigcup \overline{\boldsymbol{P}} \right\} \tag{6.12}$$

Individuals from $\mathcal{F}_i$ are repeatedly copied to the archive $\overline{\boldsymbol{P}}$ with $i = -1, 0, 1, 2, \ldots$ until the size of the archive $\overline{\boldsymbol{P}}$ equals to or exceeds N. If the size of the archive $\overline{\boldsymbol{P}}$ equals to N, the environmental selection finishes. If the archive size is larger than N, individuals in the last front $\mathcal{F}_l$, being copied to the archive $\overline{\boldsymbol{P}}$, are removed from the archive $\overline{\boldsymbol{P}}$ to reduce the size of the archive $\overline{\boldsymbol{P}}$ to N and the last front $\mathcal{F}_l$

must satisfy following conditions:

$$\sum_{i=-1}^{l-1} |\mathcal{F}_i| < N \tag{6.13}$$

$$\sum_{i=-1}^{l-1} |\mathcal{F}_i| + |\mathcal{F}_l| > N \tag{6.14}$$

Individuals in $\mathcal{F}_l$ are sorted by their crowding values. The individual with the highest crowding value in $\mathcal{F}_l$ is removed from the archive $\overline{\boldsymbol{P}}$ and from $\mathcal{F}_l$. The crowding value of the remaining individuals in $\overline{\boldsymbol{P}}$, not just $\mathcal{F}_l$, are updated accordingly. These processes are repeated until the size of the archive $\overline{\boldsymbol{P}}$ equals to N.

6.1.4 Crowding Assignment

As NSGA2 and SPEA2, HVEA employs the distance-based approach to estimate the density of individuals around a particular individual. However HVEA's crowding assignment mechanism is different from NSGA2 and SPEA2. Both NSGA2 and SPEA2 only consider "one neighbour" in the neighbouring area in order to determine the density of a particular individual. NSGA2 combines the distance to the adjacent neighbour in each objective to estimate the crowding of a particular individual, whereas SPEA2 considers the distance to the k-th nearest neighbour. HVEA takes into account all individuals in the *neighbouring area* to determine the crowding of a particular individual. The *neighbouring area* is defined by the current state of the combined population $\boldsymbol{P} \bigcup \overline{\boldsymbol{P}}$ and a *radius* ω. An individual $\vec{x^*}$ is in the *neighbouring area* of an individual $\vec{x}$ (i.e. $\vec{x^*}$ and $\vec{x}$ are neighbours) if the following condition is satisfied:

$$|x_i - x_i^*| \leq \left(r_i^{\max} - r_i^{\min}\right) \times \omega \tag{6.15}$$

for $\forall i = 1, 2, \ldots, m$ where $r_i^{\min}$ and $r_i^{\max}$ are given by equation (6.5) and (6.6) respectively. Let $\boldsymbol{NB}(\vec{x})$ be the set of all neighbouring individuals of $\vec{x}$, this means $\vec{x^*} \in \boldsymbol{NB}(\vec{x})$ if and only if $\vec{x^*}$ and $\vec{x}$ satisfy (6.15). Then, the crowding value of $\vec{x}$ is as follows:

$$crowding(\vec{x}) = \sum_{\vec{x^*} \in \boldsymbol{NB}(\vec{x})} (d(\vec{x}, \vec{x^*}) + 1)^{-1} \tag{6.16}$$

where $d(\vec{x}, \vec{x^*})$ is the Euclidean distance between $\vec{x}$ and $\vec{x^*}$

$$d(\vec{x}, \vec{x^*}) = \sqrt{\sum_{i=1}^{m} (x_i - x_i^*)^2} \tag{6.17}$$

and a lower value of $crowding(\vec{x})$ is preferred because $(\vec{x})$ is further away from other individuals in its neighbourhood.

Apart from considering all individuals in the *neighbouring area* to estimate the crowding of a particular individual, another main difference between the crowding assignment mechanism in HVEA and that of NSGA2 and SPEA2 is follows. NSGA2 only uses each front separately to estimate the crowding value of individuals in that front. SPEA2 uses all individuals in both the offspring population $\boldsymbol{P}$ and the archive $\overline{\boldsymbol{P}}$ to estimate the crowding value of individuals. However to estimate the crowding value, HVEA only uses individuals in fronts, which are already added to the archive $\overline{\boldsymbol{P}}$. Additionally, the crowding value of individual $\vec{x} \in \overline{\boldsymbol{P}}$ is repeatedly adjusted when a new front $\mathcal{F}_i$ is added to $\overline{\boldsymbol{P}}$ which is not the case in SPEA2. The pseudocode for the crowding assignment of front $\mathcal{F}_i$ $(-1 \leq i \leq l)$ is as follows:

```
Procedure CrowdingAssignment(F_i)
begin
  for each x in F_i
    for each x* in F_i (x* ≠ x)
      if x* and x are neighbours
      then add (d(x, x*) + 1)^-1 to crowding(x) and crowding(x*)
      endif
    endfor
```

```
      for each front F_j (-1 ≤ j < i)
        for each x*⃗ in F_j
          if x*⃗ and x⃗ are neighbours
          then add (d(x⃗, x*⃗) + 1)^-1 to crowding(x⃗) and crowding(x*⃗)
          endif
        endfor
      endfor
    endfor
  end
```

6.1.5 Generating Offspring

HVEA chooses parents for the mating pool using binary tournament selection. Individuals that compete to be selected as the second parent are different from those that compete to be the first parent. The binary tournament selection prioritises the following order: *ranking, crowding, fitness* values of individuals. Crossover and mutation are applied on the mating pool to form the offspring population $\boldsymbol{P}$. Any duplicated offspring dies out of the combined population $\boldsymbol{P} \bigcup \overline{\boldsymbol{P}}$. It guarantees no duplication in the combined population $\boldsymbol{P} \bigcup \overline{\boldsymbol{P}}$ before the environmental selection process. Consequently, there is no duplication of individuals in the archive $\overline{\boldsymbol{P}}$.

To summarise, HVEA requires two parameters the size of each ranking level ($0 < \mu \leq 1$) and the radius of the neighbouring area ($0 < \omega \leq 1$).

6.1.6 Comparison With SMS-EMOA

It is emphasised that HVEA is clearly different from SMS-EMOA which uses the exclusive hypervolume contribution of individuals in the worst front to eliminate an individual during the steady-state selection scheme. HVEA uses the hypervolume of an individual to determine its fitness and its ranking. The next archive is selected from the combining offspring population and archive population using individual ranking and crowding.

6.2 Comparative Case Study

6.2.1 Evolutionary Multiobjective Algorithms

The performance of HVEA is compared to that of various EMO algorithms which deploy different concepts and approaches. These EMO algorithms are NSGA2 [47], SPEA2 [124], SEAMO2 [96], IBEA [123] and MOEA/D [121]. NSGA2 and SPEA2 are two very well-known and good performing EMO algorithms which have been cited by a large number of publications since their proposals. SEAMO2 is a steady-state population and selection EMO algorithm which is simple but quite effective. IBEA is based on a new concept, solution's quality indicator, to guide the search. There are 2 quality indicators proposed by Zitzler and Künzli [123], the binary additive ϵ-indicator and the hypervolume indicator. HVEA is compared to IBEA using each of these two quality indicators. Finally, MOEA/D, a very recent proposal, decomposes a MOP into several scalar optimisation subproblems and optimises them simultaneously. There are also 2 methods to calculate the individual fitness for MOEA/D. Both these methods are implemented in this comparative study. The detail for each EMO algorithm was given in section 3.2.2 of Chapter 3.

6.2.2 Experimental Design

6.2.2.1 Multiobjective Optimisation Problem

The multiobjective optimisation problem used for investigating the performance of HVEA and other EMO algorithms is the multiple 0/1 knapsack problem proposed by Zitzler and Thiele [125]. A detailed description of the problem is given in section 3.3.1 of Chapter 3. There are 9 instances for this problem [125]. The population size used for each instance and for each EMO algorithms is given in table 6.1.

Table 6.1: Parameter Setting for The Multiple 0/1 Knapsack Problem

Instance	Population Size (S) for NSGA2, SPEA2, SEAMO2, IBEA, HVEA	Population Size (N) for MOEA/D
ks2_250	150	150
ks2_500	200	200
ks2_750	250	250
ks3_250	200	351
ks3_500	250	351
ks3_750	300	351
ks4_250	250	455
ks4_500	300	455
ks4_750	350	455

As aforementioned in section 3.3.1 there are a number of different approaches in modelling the multiple 0/1 knapsack problem. The experimental results reported here, consider the different implementations for the multiple 0/1 knapsack problem model. That is, in order to eliminate any bias due to the modelling method used to represent the multiple 0/1 knapsack problem, the performance of HVEA and selected EMO algorithms is assessed using different modelling methods separately. There are 4 different modelling-greedy-repair methods for the multiple 0/1 knapsack problem as discussed in section 3.3.1.

The assessment is based on 50 independent runs for statistical analysis, each run consists of 2000 generations. It is noticed that MOEA/D predefines the set of even spread weight vector $\{\lambda^1, \ldots, \lambda^N\}$ where $N = C_{H+m-1}^{m-1}$ (H: controlling parameter). Therefore, it is problematic to alter the population size N for MOEA/D to the required population size S. However, the same or at least nearly the same number of fitness evaluations ($2000 \times S$) should be used for comparison. Therefore, it is suggested not to alter the population size N set by MOEA/D but instead to alter the number of generations for MOEA/D as follows:

$$g = \left\lfloor \frac{2000 \times S}{N} \right\rfloor \tag{6.18}$$

6.2.2.2 Performance Metrics

The performance of HVEA and the other selected EMO algorithms is assessed based on 3 performance metrics: the hyper volume measurement proposed by Zitzler and Thiele [125], the generational distance and the inverted generational distance measurements. Additionally, the computational time spent by each EMO algorithm in these experiments is also reported.

Regarding the hypervolume metric, Zitzler and Thiele suggested that the reference point to estimate the hyper volume should be located in the origin of the objective space. However, this gives extreme points in the objective space much higher exclusive contribution to the hyper volume metric especially when the objective values of the solutions are high. Therefore, it is suggested that the reference point $\vec{r} = (r_1, r_2, \ldots, r_m)$ should be close to the solution set $\overline{\boldsymbol{P}}$ obtained by all algorithms. The estimation of the reference point is as follows:

$$u_i = \max_{\vec{x} \in \overline{\boldsymbol{P}}} \{f_i(\vec{x})\} \tag{6.19}$$

$$l_i = \min_{\vec{x} \in \boldsymbol{S}} \{f_i(\vec{x})\} \tag{6.20}$$

$$r_i = l_i - (u_i - l_i) \times 0.1 \tag{6.21}$$

$\vec{u}$ and $\vec{l}$ are referred as the upper bound and lower bound respectively for the solution set $\overline{\boldsymbol{P}}$ obtained by all algorithms. In other words, $\vec{u}$ is the least vector that dominates all solutions in $\overline{\boldsymbol{P}}$ in the objective space and $\vec{l}$ the most vector that is dominated by all solutions in $\overline{\boldsymbol{P}}$ in the objective space.

Regarding the generational and inverted generational distance metrics, the true Pareto front $\boldsymbol{P}_t$ is not available for every instance of the multiple 0/1 knapsack problem. Therefore, it is suggested (as in MOEA/D [121]) to use the approximation of $\boldsymbol{P}_t$ estimated by Jaszkiewicz [72] to determine the generational distance and inverted generational distance measurements.

6.3 Results and Discussion

Regarding HVEA, the parameter μ in equation (6.11), which determines the rank of an individual based on its fitness, is set to $\mu = 0.01$. Different *neighbouring area radius* values ω, which is given in (6.15), are experimented. The value of ω should lie in the range $0 < \omega \leq 1$. The reported results for $\omega = 0.01$ and $\omega = 1.0$ are abbreviated as hv1 and hv100 respectively in both sets of figures and tables and as $\text{HVEA}_{0.01}$ and $\text{HVEA}_{1.0}$ respectively in the text.

Other values of the size of each front $\mu = 0.025, 0.05, 0.075, 0.1, 0.15, 0.2, 0.3, 0.5$ and the *neighbouring area radius* $\omega = 0.05, 0.10, 0.15, 0.25, 0.50, 0.75$ are also investigated. However, results obtained from these combinations of the two parameters μ and ω are much worse than these ones reported here ($\mu = 0.01$ and $\omega = 0.01, 1.0$). The results are worse because larger values of μ lead to the ranks for solutions in the population being the same which is not helpful to classify solutions into different fronts. Regarding the *neighbouring area radius*, $\omega = 0.01$ indicates a very small neighbouring area and $\omega = 1.0$ indicates that every other solutions in the population are neighbours of a solution. These two values of ω provide a more consistent evaluation for each solution in terms of assessing its neighbouring area.

6.3.1 Comparison to EMO algorithms

The performance of HVEA is compared to NSGA2, SEAMO2, SPEA2, $\text{IBEA}_{\epsilon+}$ (additive ϵ-indicator) and IBEA_{HV} (hypervolume indicator) abbreviated as ns2, sea2, sp2, $\text{ib}_{\epsilon+}$ (ibe) and ib_{HV} (ibhv) repsectively. The performance of these EMO algorithms is assessed using four greedy repair methods for the multiple 0/1 knapsack problem: the *binary encoding* approach proposed by Zitzler and Thiele [125], the *permutation encoding* approach proposed by Mumford [115], and the 2 different weighted linear scalarising approaches proposed by Jaszkiewicz [72], namely *te binary encoding* (Tchebycheff) and *ws binary encoding* (weighted sum).

Figures 6.2 to 6.5 represent the boxplots of the dominated space (the hypervolume $\mathcal{S}$-metric value) for each algorithm. The higher (vertically) the boxplot is, the better performance of the algorithm is. Tables 6.2 to 6.5 and tables 6.6 to 6.9 represent the average generational distance and the average inverted generational distance from the obtained Pareto fronts to the (estimated) Pareto-optimal front. The lower value of the distance is, the better performance of the algorithm is. Finally, table 6.10 to 6.13 indicate the average computational time (in seconds) for each algorihms. The best value is highlighted as bold in all tables.

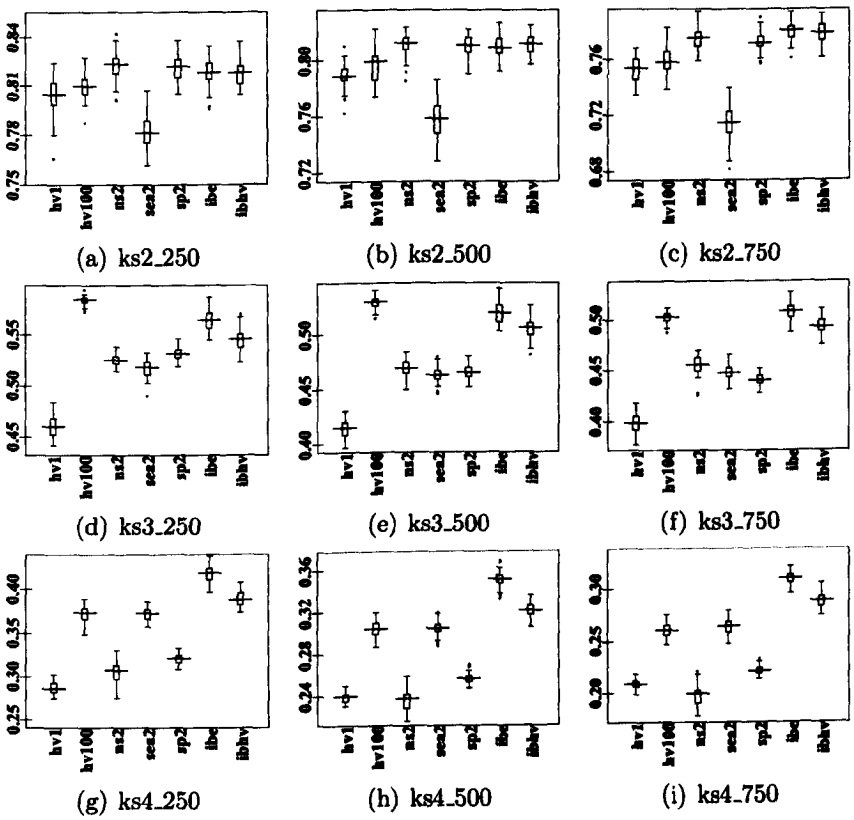
Regarding the hypervolume metric, Figures 6.2 to 6.5 show that $\text{HVEA}_{1.0}$ outperforms or at least remains competitive to NSGA2, SEAMO2, SPEA2 on all 9 instances of the multiple 0/1 knapsack problem using all 4 different greedy repair methods. $\text{HVEA}_{1.0}$ is worse in only few cases out of 36 instance-repair method combinations which are mainly in the lowest dimension problem, that one with 2-knapsacks. $\text{HVEA}_{1.0}$ outperforms IBEA (both $\text{IBEA}_{\epsilon+}$ and IBEA_{HV}) for the 3-knapsack problems but it is less competitive on the 2- and 4- knapsack problems. $\text{IBEA}_{\epsilon+}$ tends to be the most effective EMO algorithm to provide the coverage (hypervolume) for the Pareto fronts. It is also noticed that $\text{HVEA}_{1.0}$ performs much better than $\text{HVEA}_{0.01}$. The reason for this is that $\text{HVEA}_{0.01}$ only looks at a tiny neighbouring area to determine the *crowding* of an individual whereas $\text{HVEA}_{1.0}$ examines the whole objective space, i.e. every individual contributing towards computing the *crowding* of an individual. Results on the generational and inverted generational distance show more prominent evidence regarding this issue.

Tables 6.2 to 6.5 show the generational distance from the non-dominated sets found to the true Pareto front while Tables 6.6 to 6.9 show the inverted generational distance from the true Pareto front to the non-dominated sets found. Bold figures indicate the best results for each instance. Both sets of tables clearly indicate that $\text{HVEA}_{1.0}$ provides better diversity than $\text{HVEA}_{0.01}$ whereas $\text{HVEA}_{0.01}$ provides better convergence than $\text{HVEA}_{1.0}$ due to the use of *neighbouring area radius* ω. On

the performance regarding the generational distance metric, $HVEA_{0.01}$ outperforms NSGA2 and SPEA2 and is competitive to SEAMO2 and IBEA. SEAMO2 is one of the best algorithms in terms of convergence, although SEAMO2 only exploits Pareto dominance without fitness, ranking and crowding estimation. The reason behind the good convergence ability of SEAMO2 is that this is a steady-state algorithm which allows offspring to compete for recombination immediately within the current generation. However, this also leads SEAMO2 to be one of the worst EMO algorithms in terms of diversity in the non-dominated sets found. The inverted generational distance metric shows that $HVEA_{1.0}$ outperforms other EMO algorithms in most cases of the 36 instance-repair method combinations. $HVEA_{1.0}$ is clearly the best algorithm to provide diversity in high dimensional problems, the ones with 3- and 4-knapsacks. There is no strong evidence to determine which algorithm is the best in the 2-knapsack problem.

The computational time spent by each algorithm are summarised in Tables 6.10 to 6.13. We can see that SPEA2 and IBEA are the slowest algorithms due to the complexity in computation of fitness values (SPEA2) and indicator values (IBEA) while SEAMO2, NSGA2, HVEA are much faster algorithms.

Taken all the above criteria into account, it is concluded that HVEA outperforms NSGA2, SEAMO2, and SPEA2. The proposed algorithm remains competitive to IBEA but it is much faster in terms of computational time and this is an important advantage particularly in high dimensional problems.

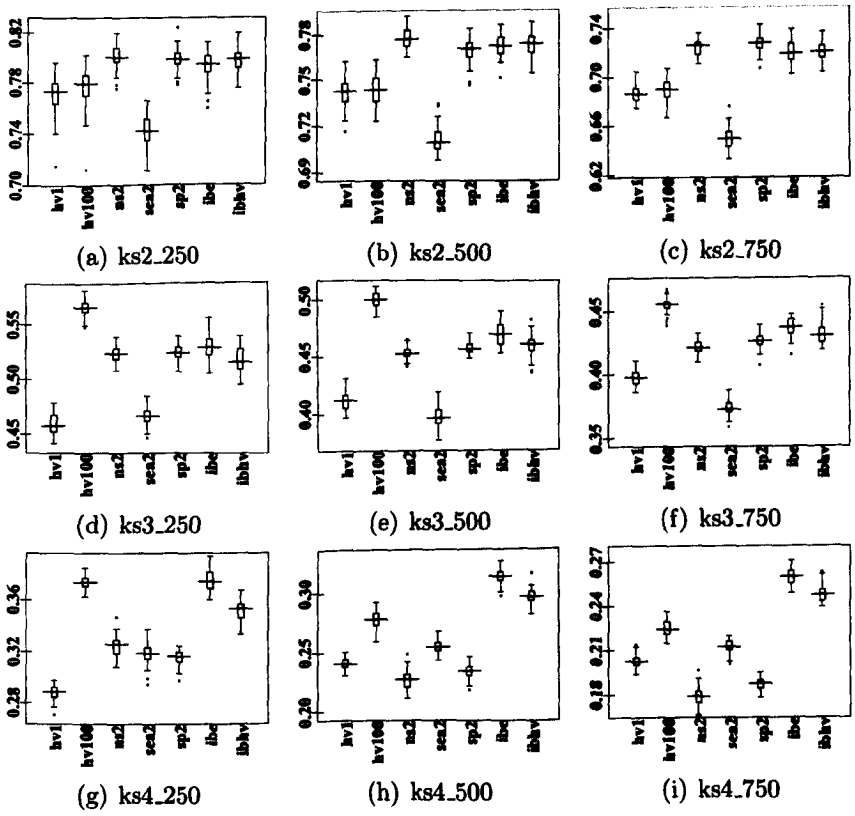


(a) ks2_250 (b) ks2_500 (c) ks2_750

(d) ks3_250 (e) ks3_500 (f) ks3_750

(g) ks4_250 (h) ks4_500 (i) ks4_750

Figure 6.2: The hypervolume $\mathcal{S}$-metric (permutation encoding)

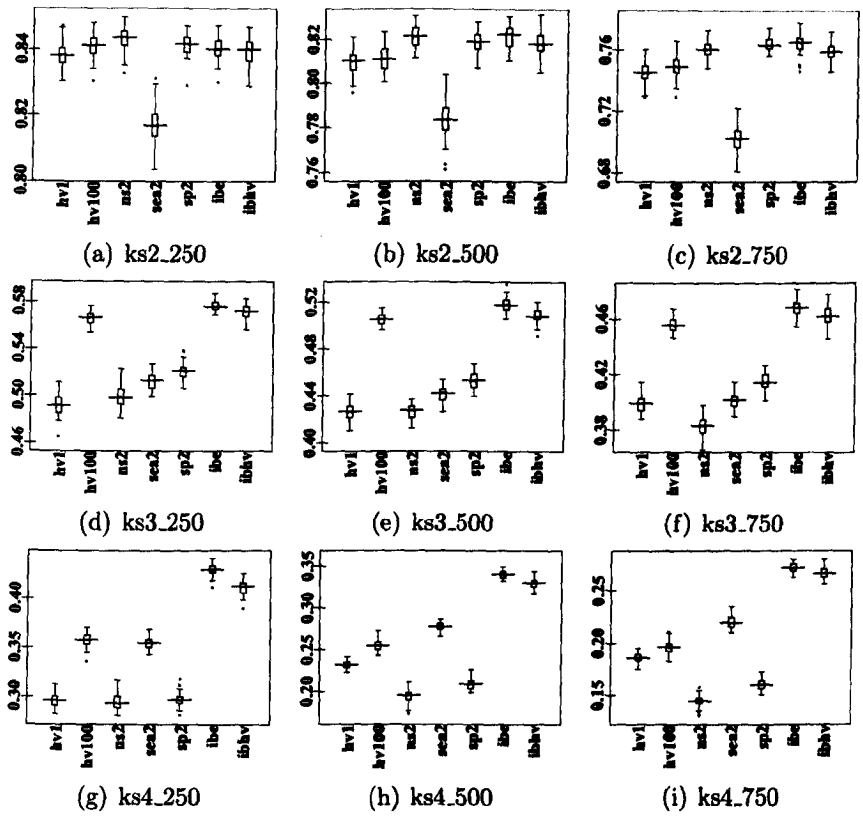


(a) ks2_250 (b) ks2_500 (c) ks2_750

(d) ks3_250 (e) ks3_500 (f) ks3_750

(g) ks4_250 (h) ks4_500 (i) ks4_750

Figure 6.3: The hypervolume $\mathcal{S}$-metric (binary encoding)

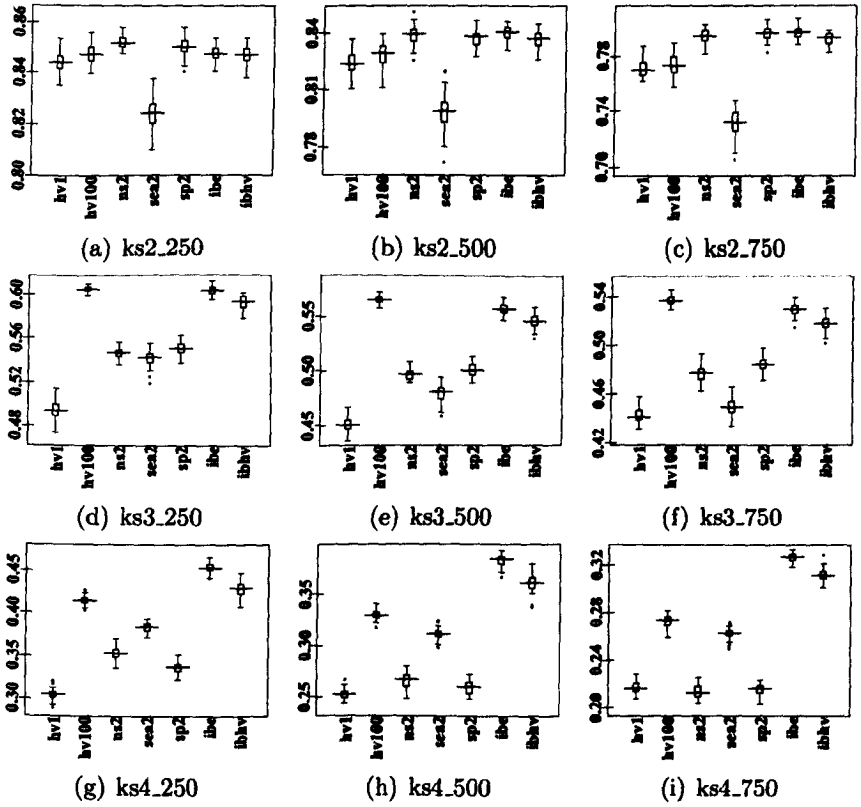


(a) ks2_250 (b) ks2_500 (c) ks2_750

(d) ks3_250 (e) ks3_500 (f) ks3_750

(g) ks4_250 (h) ks4_500 (i) ks4_750

Figure 6.4: The hypervolume $\mathcal{S}$-metric (te binary encoding)



(a) ks2_250 (b) ks2_500 (c) ks2_750

(d) ks3_250 (e) ks3_500 (f) ks3_750

(g) ks4_250 (h) ks4_500 (i) ks4_750

Figure 6.5: The hypervolume $\mathcal{S}$-metric (ws binary encoding)

Table 6.2: Generational Distance (permutation encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	3.20	3.24	1.99	4.57	**1.74**	3.31	3.45
ks2_500	10.10	10.37	6.00	14.47	**5.46**	10.65	12.02
ks2_750	17.86	17.48	12.73	26.54	**11.36**	21.86	23.71
ks3_250	6.96	14.59	24.31	8.04	10.83	6.15	**5.73**
ks3_500	16.68	34.11	54.61	18.24	18.04	13.91	**12.75**
ks3_750	26.57	54.02	73.69	29.58	23.35	22.61	**20.36**
ks4_250	11.43	38.73	44.97	12.60	19.93	12.27	**11.39**
ks4_500	22.37	66.97	95.02	27.28	27.74	22.97	**20.84**
ks4_750	34.44	99.30	141.16	42.77	34.04	35.25	**30.96**

Table 6.3: Generational Distance (binary encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	3.91	**3.78**	4.17	6.16	4.09	6.95	6.99
ks2_500	11.42	**11.03**	13.39	15.10	13.78	17.66	18.07
ks2_750	**26.86**	27.10	32.87	33.91	31.67	29.48	30.05
ks3_250	**8.45**	13.95	22.16	8.62	14.18	10.06	9.77
ks3_500	**20.83**	34.04	50.69	21.39	32.63	24.47	22.82
ks3_750	**35.71**	55.08	74.63	40.78	52.56	38.58	35.84
ks4_250	14.18	33.23	38.58	**13.91**	27.10	15.37	14.99
ks4_500	31.13	75.38	96.31	30.97	66.35	32.20	**28.85**
ks4_750	52.80	119.03	150.58	52.89	106.08	53.40	**44.81**

Table 6.4: Generational Distance (te binary encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	**4.42**	4.46	5.18	6.94	5.41	6.84	6.93
ks2_500	**16.33**	16.60	17.68	23.92	17.62	18.35	19.01
ks2_750	44.67	43.98	46.88	64.90	44.96	**41.62**	43.42
ks3_250	13.99	18.92	28.85	**11.79**	21.06	13.64	13.50
ks3_500	36.35	44.72	70.27	33.93	50.40	31.61	**29.84**
ks3_750	64.61	78.85	113.39	67.64	85.45	59.47	**56.29**
ks4_250	22.82	40.94	47.44	**16.53**	37.85	17.53	17.29
ks4_500	51.52	89.40	114.67	40.71	89.75	43.05	**37.09**
ks4_750	90.24	144.94	183.81	79.45	148.89	78.48	**67.98**

Table 6.5: Generational Distance (ws binary encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	2.02	**1.96**	2.24	3.89	2.07	2.91	2.92
ks2_500	6.13	**6.07**	6.70	10.80	6.74	8.07	8.59
ks2_750	16.53	**16.38**	18.42	27.96	17.26	17.40	17.62
ks3_250	9.52	12.17	20.82	**7.26**	14.23	7.87	7.46
ks3_500	20.87	27.25	47.72	16.50	31.60	16.88	**15.90**
ks3_750	30.72	42.91	67.05	25.88	45.73	27.35	**24.99**
ks4_250	17.06	29.66	35.52	**12.52**	27.13	13.59	13.35
ks4_500	33.58	60.66	82.87	25.34	59.50	27.40	**23.68**
ks4_750	54.58	97.91	131.27	42.68	96.22	44.91	**38.29**

Table 6.6: Inverted Generational Distance (permutation encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	10.13	9.06	**8.45**	11.49	8.67	8.48	8.48
ks2_500	10.96	9.98	10.07	13.93	9.86	9.50	**8.62**
ks2_750	48.07	46.02	42.92	64.14	44.40	38.07	**38.01**
ks3_250	21.11	**5.59**	10.37	18.35	12.41	13.25	15.74
ks3_500	48.73	**15.58**	25.60	43.54	39.14	33.71	37.69
ks3_750	69.65	**27.06**	39.73	61.16	61.03	48.20	54.06
ks4_250	19.33	**9.39**	13.57	14.98	14.53	12.30	15.03
ks4_500	43.24	**19.71**	30.15	33.84	39.64	29.83	35.81
ks4_750	68.36	**33.56**	49.35	54.58	65.93	49.84	57.93

Table 6.7: Inverted Generational Distance (binary encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	14.55	13.88	**9.75**	16.81	10.51	10.67	9.89
ks2_500	17.97	17.91	**11.75**	20.95	12.85	12.19	11.79
ks2_750	81.69	81.09	54.68	98.40	**54.63**	56.50	55.30
ks3_250	20.33	**6.76**	10.20	20.46	11.16	14.33	16.36
ks3_500	46.55	**22.41**	27.22	49.62	31.86	36.09	39.43
ks3_750	64.04	**40.16**	43.82	72.18	48.56	53.63	56.87
ks4_250	18.37	**8.47**	12.53	16.64	12.67	13.45	15.90
ks4_500	39.55	**22.05**	30.54	37.02	30.33	29.45	34.11
ks4_750	62.33	**38.87**	51.83	60.39	51.45	49.51	55.66

Table 6.8: Inverted Generational Distance (te binary encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	3.06	2.93	**2.62**	4.52	2.87	2.84	2.89
ks2_500	6.88	6.80	5.79	8.72	6.03	**5.52**	5.72
ks2_750	44.16	42.82	41.38	56.88	39.69	**38.89**	40.27
ks3_250	14.83	**7.03**	12.19	17.76	9.19	9.10	11.10
ks3_500	37.33	**19.40**	30.71	40.87	26.50	23.96	27.44
ks3_750	52.74	**36.67**	52.38	59.90	42.18	37.78	41.14
ks4_250	15.02	10.27	14.10	14.49	11.77	**8.90**	11.58
ks4_500	35.20	25.91	35.61	32.69	30.65	**22.04**	26.31
ks4_750	56.33	45.72	60.56	53.05	52.58	**37.70**	43.18

Table 6.9: Inverted Generational Distance (ws binary encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	3.80	3.40	**2.70**	4.90	3.05	3.14	3.34
ks2_500	7.02	6.62	4.97	9.05	5.27	**4.53**	4.78
ks2_750	41.14	40.31	28.86	52.55	27.86	**26.11**	27.82
ks3_250	16.83	**4.78**	9.71	17.31	8.93	9.06	11.05
ks3_500	40.25	**13.21**	22.43	40.48	24.74	23.92	27.69
ks3_750	56.43	**23.72**	35.38	59.39	37.02	36.85	41.70
ks4_250	16.08	**7.75**	11.47	14.33	11.35	9.17	11.85
ks4_500	37.35	**18.10**	27.27	32.41	28.35	22.44	27.55
ks4_750	58.56	**31.92**	46.15	52.21	47.00	38.13	45.26

Table 6.10: Computational Time in seconds (permutation encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	7	6	6	**4**	48	50	50
ks2_500	14	14	14	**9**	97	96	91
ks2_750	27	28	26	**21**	165	160	144
ks3_250	14	18	11	**7**	103	103	158
ks3_500	28	38	23	**15**	179	175	242
ks3_750	46	66	39	**32**	272	264	368
ks4_250	24	52	18	**11**	184	183	354
ks4_500	44	114	36	**22**	294	286	545
ks4_750	68	204	56	**48**	415	407	771

Table 6.11: Computational Time in seconds (binary encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	8	8	7	**6**	50	51	51
ks2_500	18	18	17	**14**	99	96	95
ks2_750	31	32	30	**26**	165	157	149
ks3_250	16	20	12	**9**	104	103	145
ks3_500	31	39	25	**21**	175	170	236
ks3_750	50	61	43	**36**	266	256	350
ks4_250	27	50	19	**14**	185	183	351
ks4_500	47	107	37	**28**	281	278	537
ks4_750	73	168	60	**48**	396	392	746

Table 6.12: Computational Time in seconds (te binary encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	13	13	12	**10**	54	55	60
ks2_500	31	31	30	**26**	112	109	117
ks2_750	58	58	56	**51**	192	182	198
ks3_250	22	28	18	**16**	109	110	150
ks3_500	51	61	46	**40**	195	191	256
ks3_750	93	109	86	**76**	310	301	392
ks4_250	36	66	30	**24**	195	194	367
ks4_500	75	143	66	**57**	307	308	565
ks4_750	134	256	129	**108**	462	461	810

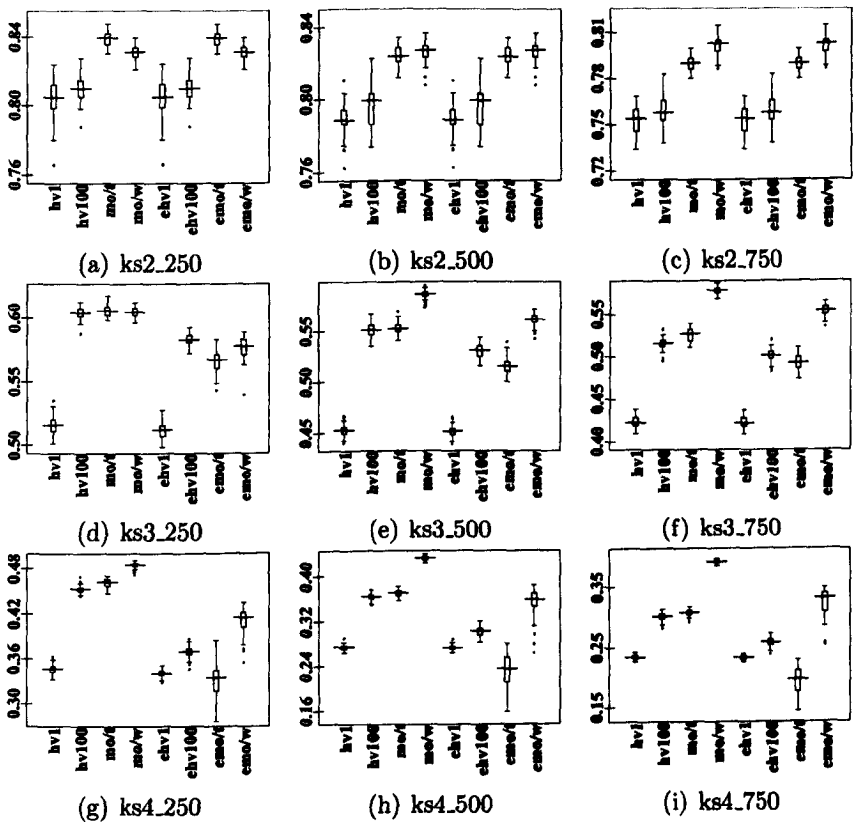
Table 6.13: Computational Time in seconds (ws binary encoding)

Instance	hv1	hv100	ns2	sea2	sp2	$ib_{\epsilon+}$	ib_{HV}
ks2_250	12	13	12	**9**	54	54	53
ks2_500	29	30	28	**24**	111	106	100
ks2_750	55	55	52	**48**	188	178	164
ks3_250	22	27	18	**15**	108	109	162
ks3_500	50	60	43	**38**	192	190	266
ks3_750	87	103	80	**71**	302	295	390
ks4_250	36	67	29	**22**	194	193	367
ks4_500	71	135	62	**53**	304	304	563
ks4_750	122	233	114	**95**	449	446	800

6.3.2 Comparison to MOEA/D

The performance of HVEA is also compared to MOEA/D separately due to the very particular settings in the approach employed by MOEA/D. As discussed in section 6.2.2.1, it is problematic to alter the population size N in MOEA/D in accordance with the population size S suggested for the multiple 0/1 knapsack problem. Furthermore, MOEA/D stores all non-dominated solutions found so far in an external archive which is then reported as the final result. Therefore, it is suggested to adapt HVEA to the setting similar to MOEA/D in terms of the population size and the external archive for non-dominated solutions. Experimental results, including figures and tables, are presented in a similar format as in section 6.3.1. MOEA/D employed Tchebycheff and weight sum approach for the fitness calculation and these variants are abbreviated as "mo/t" and "mo/w" in the results presented below. The extracted population (truncated external population) is also reported so that the size of the extracted population does not exceed the suggested population size S. The truncation approach of NSGA2 is used here. In both sets of figures and tables, the extracted populations are presented using prefix "e", e.g. ehv1 corresponds to the extracted population for algorithm hv1.

We can see in the results presented in Figures 6.6 to 6.9 and Tables 6.14 to 6.21 that MOEA/D outperforms HVEA in most cases out of 36 instance-repair method combinations on the hypervolume metric and the inverted generational distance metric. The convergence of MOEA/D is only better than that of HVEA in the linear scalarising greedy repair method. However the computational time of MOEA/D is considerably higher than that of HVEA. In the other 2 repair methods, $\text{HVEA}_{0.01}$ outperforms MOEA/D regarding the generational distance metric. Under a restriction of the size of the population, where the truncation is applied to the external population, HVEA is able to compete with MOEA/D. However HVEA is much faster than MOEA/D as it can be seen in Tables 6.23 to 6.25.

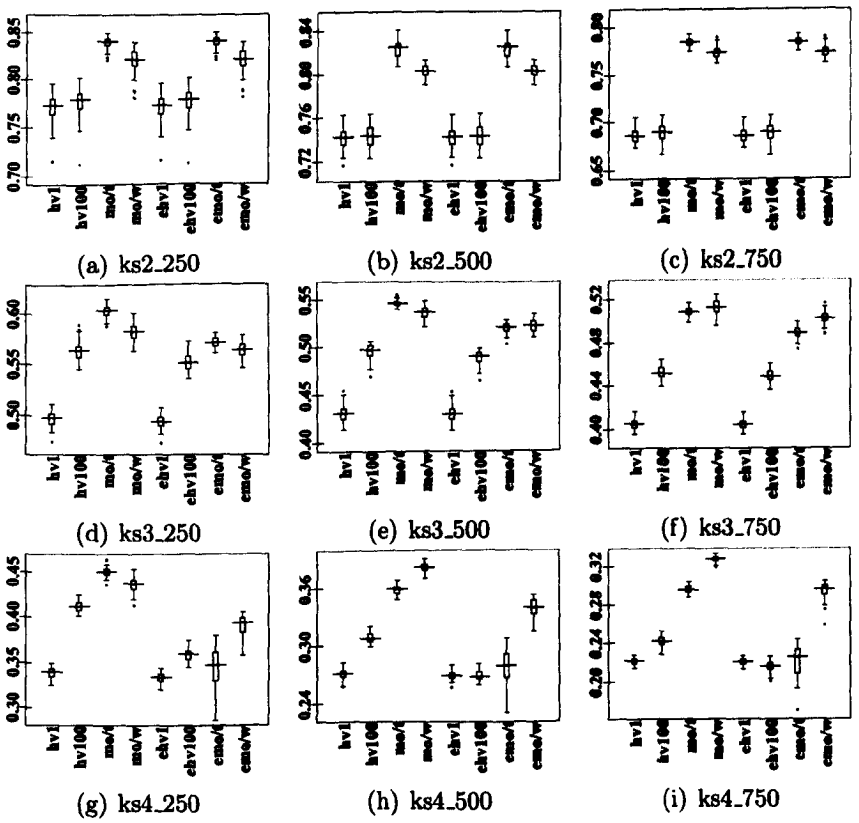


(a) ks2_250 (b) ks2_500 (c) ks2_750

(d) ks3_250 (e) ks3_500 (f) ks3_750

(g) ks4_250 (h) ks4_500 (i) ks4_750

Figure 6.6: The hypervolume $\mathcal{S}$-metric (permutation encoding)

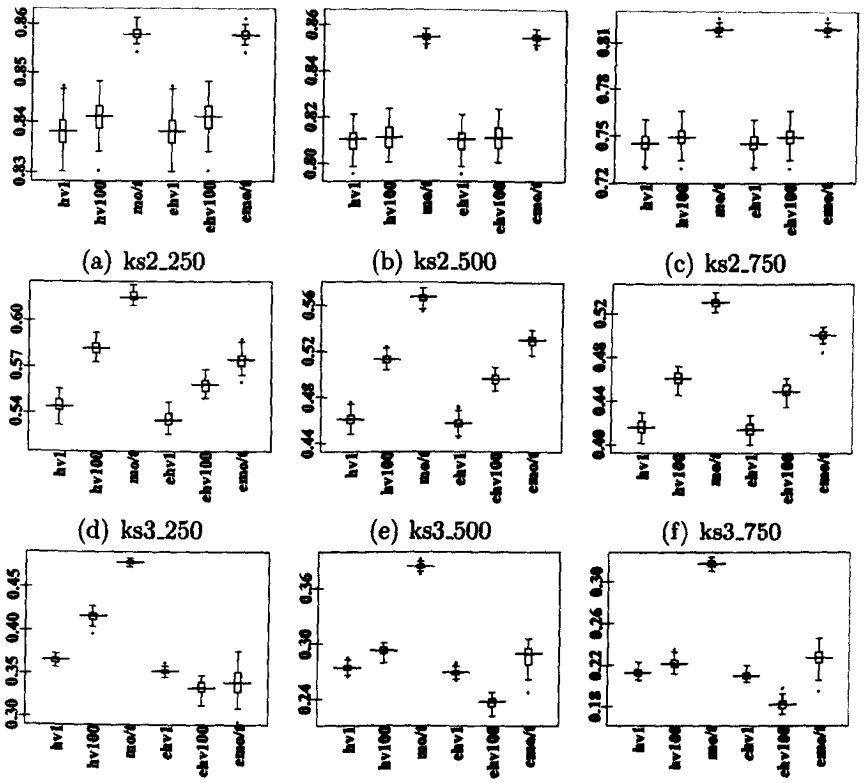


(a) ks2_250 (b) ks2_500 (c) ks2_750

(d) ks3_250 (e) ks3_500 (f) ks3_750

(g) ks4_250 (h) ks4_500 (i) ks4_750

Figure 6.7: The hypervolume $\mathcal{S}$-metric (binary encoding)

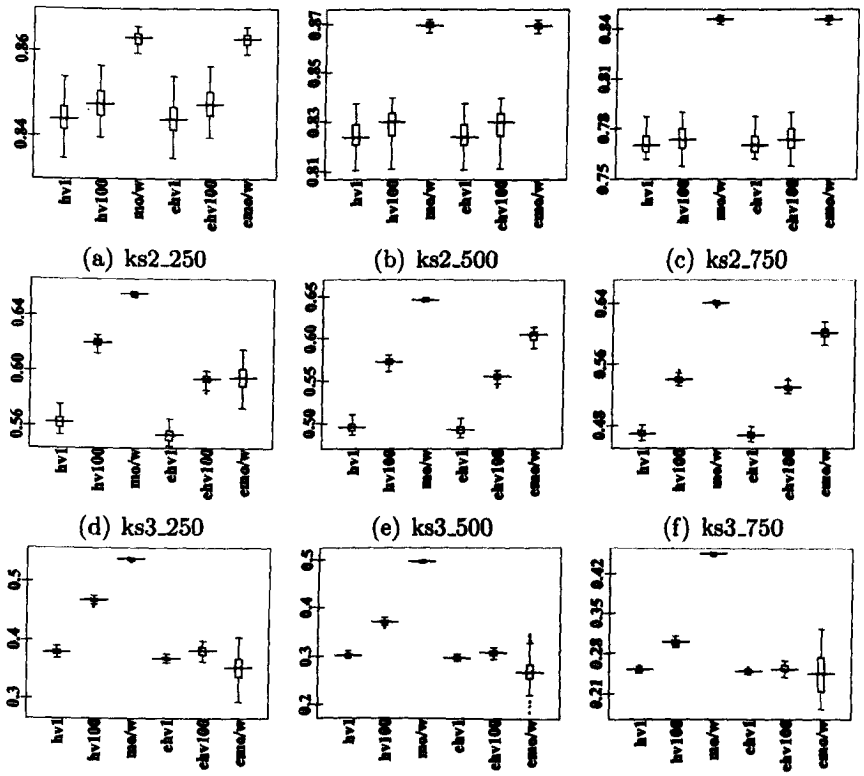


(a) ks2_250 (b) ks2_500 (c) ks2_750

(d) ks3_250 (e) ks3_500 (f) ks3_750

(g) ks4_250 (h) ks4_500 (i) ks4_750

Figure 6.8: The hypervolume $\mathcal{S}$-metric (te binary encoding)



(a) ks2_250 (b) ks2_500 (c) ks2_750

(d) ks3_250 (e) ks3_500 (f) ks3_750

(g) ks4_250 (h) ks4_500 (i) ks4_750

Figure 6.9: The hypervolume $\mathcal{S}$-metric (ws binary encoding)

Table 6.14: Generational Distance (permutation encoding)

Instance	hv1	hv100	mo/t	mo/w	ehv1	ehv100	emo/t	emo/w
ks2_250	**3.16**	3.21	5.98	7.00	**3.19**	3.23	6.18	7.18
ks2_500	**10.00**	10.17	17.57	16.05	**10.08**	10.33	18.64	16.84
ks2_750	17.49	**17.06**	29.28	24.43	17.78	**17.40**	33.04	26.86
ks3_250	**2.08**	2.84	4.64	3.86	**7.99**	9.90	14.77	13.18
ks3_500	**4.97**	6.81	10.31	6.84	**16.72**	27.44	37.67	26.73
ks3_750	**9.29**	12.21	17.22	9.99	**26.94**	48.53	59.89	38.67
ks4_250	**2.24**	3.75	3.80	2.92	**14.11**	30.34	20.63	17.17
ks4_500	**4.23**	6.79	8.21	5.11	**24.22**	57.09	39.98	34.94
ks4_750	**6.79**	10.63	13.32	7.47	**34.17**	86.46	61.01	51.07

Table 6.15: Generational Distance (binary encoding)

Instance	hv1	hv100	mo/t	mo/w	ehv1	ehv100	emo/t	emo/w
ks2_250	3.91	**3.78**	4.47	7.28	3.91	**3.78**	4.54	7.29
ks2_500	11.42	**11.03**	15.53	17.39	11.42	**11.03**	15.53	17.39
ks2_750	**26.86**	27.10	34.07	34.88	**26.86**	27.10	34.07	34.88
ks3_250	**3.26**	4.08	4.77	4.57	**10.18**	12.28	14.33	14.01
ks3_500	**10.67**	10.69	12.18	10.15	**24.41**	31.16	37.44	32.18
ks3_750	22.54	**22.11**	23.62	19.32	**38.93**	53.36	64.60	52.75
ks4_250	**3.26**	4.46	4.08	3.63	**16.27**	26.40	21.17	19.24
ks4_500	**9.37**	11.80	10.06	8.09	**35.47**	69.12	49.77	44.30
ks4_750	18.56	20.13	**16.93**	12.85	**57.80**	114.62	80.97	71.23

Table 6.16: Generational Distance (te binary encoding)

Instance	hv1	hv100	mo/t	ehv1	ehv100	emo/t
ks2_250	4.40	4.43	**2.83**	4.41	4.45	**3.05**
ks2_500	16.33	16.60	**9.88**	16.33	16.60	**10.03**
ks2_750	44.67	43.98	**23.54**	44.67	43.98	**23.63**
ks3_250	5.39	6.09	**4.54**	**14.62**	15.61	14.80
ks3_500	16.66	15.73	**10.49**	37.71	42.51	**37.10**
ks3_750	35.66	30.38	**19.88**	66.81	77.08	**65.21**
ks4_250	4.91	5.75	**3.61**	22.65	34.78	**20.41**
ks4_500	14.45	14.45	**8.09**	52.51	83.35	**46.14**
ks4_750	30.20	24.54	**13.47**	93.91	141.43	**65.90**

Table 6.17: Generational Distance (ws binary encoding)

Instance	hv1	hv100	mo/w	ehv1	ehv100	emo/w
ks2_250	1.91	1.85	**1.81**	2.01	**1.95**	1.96
ks2_500	6.13	6.07	**3.74**	6.13	6.07	**4.23**
ks2_750	16.53	16.38	**8.10**	16.53	16.38	**9.08**
ks3_250	2.61	3.24	**1.84**	**9.82**	10.42	10.28
ks3_500	7.64	8.11	**3.15**	21.95	25.21	**18.53**
ks3_750	14.23	14.80	**5.15**	33.44	41.42	**27.36**
ks4_250	3.02	3.66	**1.56**	17.80	24.37	**16.10**
ks4_500	7.42	9.11	**2.72**	35.33	54.99	**30.81**
ks4_750	14.49	16.17	**4.34**	58.33	93.31	**44.10**

Table 6.18: Inverted Generational Distance (permutation encoding)

Instance	hv1	hv100	mo/t	mo/w	ehv1	ehv100	emo/t	emo/w
ks2_250	10.13	9.06	**3.07**	4.26	10.13	9.06	**3.07**	4.26
ks2_500	10.96	9.98	6.60	**6.55**	10.96	9.98	6.60	**6.55**
ks2_750	48.07	46.02	37.36	**31.90**	48.07	46.02	37.37	**31.90**
ks3_250	15.86	5.22	**5.20**	6.26	15.95	**6.52**	7.83	8.37
ks3_500	42.80	**15.02**	16.43	15.10	42.85	**16.99**	19.83	18.18
ks3_750	63.84	25.89	30.90	**23.10**	63.89	27.77	34.74	**27.54**
ks4_250	14.97	**5.32**	5.76	5.91	15.08	11.06	12.70	**9.77**
ks4_500	38.54	14.94	16.82	**13.49**	38.60	22.45	31.17	**20.77**
ks4_750	63.77	29.00	31.73	**22.26**	63.87	36.84	49.64	**31.97**

Table 6.19: Inverted Generational Distance (binary encoding)

Instance	hv1	hv100	mo/t	mo/w	ehv1	ehv100	emo/t	emo/w
ks2_250	14.55	13.88	**2.78**	5.35	14.55	13.88	**2.79**	5.35
ks2_500	17.97	17.91	**5.36**	7.37	17.97	17.91	**5.36**	7.37
ks2_750	81.69	81.09	**33.08**	35.57	81.69	81.09	**33.08**	35.57
ks3_250	16.31	8.36	**5.22**	8.37	16.36	8.93	**7.41**	9.67
ks3_500	41.47	25.90	**17.05**	24.17	41.50	26.38	**20.07**	25.59
ks3_750	61.05	42.30	**30.87**	34.95	61.08	42.67	**33.83**	36.71
ks4_250	14.87	6.95	**6.10**	8.84	14.97	**10.12**	11.59	10.91
ks4_500	34.46	19.55	**17.22**	19.04	34.53	24.15	24.86	**22.39**
ks4_750	56.85	36.89	**32.02**	32.62	56.92	41.66	43.02	**36.28**

Table 6.20: Inverted Generational Distance (te binary encoding)

Instance	hv1	hv100	mo/t	ehv1	ehv100	emo/t
ks2_250	3.06	2.93	**1.52**	3.06	2.93	**1.53**
ks2_500	6.88	6.80	**3.69**	6.88	6.80	**3.69**
ks2_750	44.16	42.82	**26.22**	44.16	42.82	**26.22**
ks3_250	10.22	6.18	**5.03**	10.66	7.68	**7.45**
ks3_500	30.69	19.09	**15.73**	30.86	20.57	**18.94**
ks3_750	47.42	35.89	**31.87**	47.63	37.57	**35.11**
ks4_250	10.89	7.15	**5.52**	**11.17**	12.61	12.42
ks4_500	29.43	22.33	**16.88**	29.56	29.20	**24.87**
ks4_750	49.19	42.32	**32.32**	49.37	50.01	**42.92**

Table 6.21: Inverted Generational Distance (ws binary encoding)

Instance	hv1	hv100	mo/w	ehv1	ehv100	emo/w
ks2_250	3.80	3.39	**1.20**	3.80	3.40	**1.21**
ks2_500	7.02	6.62	**1.66**	7.02	6.62	**1.66**
ks2_750	41.14	40.31	**10.29**	41.14	40.31	**10.31**
ks3_250	10.80	4.31	**2.85**	11.13	**6.13**	8.32
ks3_500	31.98	14.04	**6.71**	32.08	15.44	**13.18**
ks3_750	48.87	24.76	**11.37**	48.95	25.93	**21.12**
ks4_250	11.45	4.99	**2.97**	11.66	**10.52**	13.93
ks4_500	30.56	14.94	**6.90**	30.66	**21.67**	31.76
ks4_750	50.08	29.11	**12.00**	50.20	**36.67**	47.92

Table 6.22: Computational Time in seconds (permutation encoding)

Instance	hv1	hv100	mo/t	mo/w
ks2_250	4	5	**3**	4
ks2_500	**8**	9	**8**	9
ks2_750	**15**	16	17	19
ks3_250	**16**	26	40	38
ks3_500	**29**	84	118	133
ks3_750	**46**	177	172	181
ks4_250	**47**	156	357	315
ks4_500	**112**	602	1025	972
ks4_750	**163**	870	1784	1807

Table 6.23: Computational Time in seconds (binary encoding)

Instance	hv1	hv100	mo/t	mo/w
ks2_250	8	8	**5**	**5**
ks2_500	18	19	**13**	**13**
ks2_750	32	33	**25**	**25**
ks3_250	25	35	**18**	**18**
ks3_500	**39**	56	**39**	40
ks3_750	**56**	74	59	57
ks4_250	**55**	139	113	87
ks4_500	**82**	285	359	342
ks4_750	**110**	427	536	538

Table 6.24: Computational Time in seconds (te binary encoding)

Instance	hv1	hv100	mo/t
ks2_250	14	13	**9**
ks2_500	33	32	**25**
ks2_750	62	59	**51**
ks3_250	26	42	**24**
ks3_500	**58**	81	60
ks3_750	**102**	126	105
ks4_250	**49**	164	98
ks4_500	**92**	331	424
ks4_750	**157**	527	769

Table 6.25: Computational Time in seconds (ws binary encoding)

Instance	hv1	hv100	mo/w
ks2_250	9	13	**8**
ks2_500	**21**	30	22
ks2_750	**39**	55	43
ks3_250	**21**	44	33
ks3_500	**41**	82	107
ks3_750	**66**	124	137
ks4_250	**47**	180	339
ks4_500	**75**	326	885
ks4_750	**111**	500	1377

6.3.3 Further Discussion

HVEA employs a parameter μ in equation (6.11) to define the size of each front. The parameter μ is set to 0.01, which is appropriate not only to the multiple 0/1 knapsack problem, but could also be used in the settings when tackling other multiobjective optimisation problems. That is, it is suggested that this parameter μ should be fixed to 0.01 rather than making it tunable. The more important parameter in HVEA is ω, the *neighbouring area radius*, which should be in the range $[0, 1]$. Section 6.3.1 shows a considerable difference in performance between $\text{HVEA}_{0.01}$ and $\text{HVEA}_{1.0}$. It was observed that $\text{HVEA}_{1.0}$ gives a much better performance regarding the hypervolume metric and the diversity of the non-dominated set. However, $\text{HVEA}_{0.01}$ is better in convergence than $\text{HVEA}_{1.0}$. One could tune ω to obtain the desirable performance. For example, ω could be set to 1.0 for benchmark problems such as the multiple 0/1 knapsack problem to obtain non-dominated set with better diversity (generational distance metric) and better coverage (hypervolume metric). However, for real-world applications, where computational time could be expensive, $\omega = 0.01$ could be deployed. Furthermore, in real-world applications, extreme solutions are likely to be of less of interest whereas better convergence and striking a good balance amongst objectives, is arguably more important. Experiments were also performed using different ω values in the range $[0, 1]$ but the performance of HVEA degrades quite significantly. Therefore, it is suggested to use $\omega = 1.0$ for better diversity and better coverage and $0.01 \leq \omega \leq 0.05$ for better convergence and faster computational time.

The performance of HVEA against other EMO algorithms is also examined using a fixed computational time, although this approach is not widely adopted in the literature due to its low reliability. The computational time (in seconds), reported in Table 6.26, is deduced from the average computational time of all EMO algorithms on each knapsack instances over 2000 generations. The computational time for the linear scalarising greedy repair method proposed by Jaszkiewicz [72] is

twice as much as the ones proposed by Mumford [115] and Zitzler and Thiele [125].

Under this computation time restriction, HVEA outperforms NSGA2, SPEA2, $IBEA_{\epsilon+}$ and $IBEA_{HV}$. HVEA only outperforms SEAMO2 on the 2- and 3-knapsack problems. SEAMO2 is slightly better than HVEA on 4-knapsack problems. It is suggested that due to the steady-state approach and the simple Pareto dominance, SEAMO2 is able to perform more evaluations than any other EMO algorithms which could lead to a better performance. HVEA is slightly worse than but competitive to MOEA/D.

Table 6.26: Computational Time (in seconds) for The Multiple 0/1 Knapsack Problem

Instance	Mumford [115] & Zitzler and Thiele [125]	Jaszkiewicz [72]
ks2_250	5	10
ks2_500	15	30
ks2_750	30	60
ks3_250	15	30
ks3_500	30	60
ks3_750	45	90
ks4_250	30	60
ks4_500	45	90
ks4_750	60	120

6.4 Summary

This chapter proposes a new population-based multiobjective evolutionary algorithm, the *Hyper Volume Evolutionary Algorithm* (HVEA). The experimental results show that HVEA outperforms or remains competitive to various state-of-the-art EMO algorithms including NSGS2, SEAMO2, SPEA2, $IBEA_{\epsilon+}$, $IBEA_{HV}$ and the strong performing MOEA/D. HVEA incorporates a new individual fitness assignment strategy using the hypervolume of an individual without the requirement of determining the reference point for the hypervolume calculation. HVEA assesses the crowding of an individual by considering all individuals in its neighbourhood.

By tuning the only parameter ω, the *neighbouring area radius*, a desired performance of the non-dominated set is attainable. That is, depending on the setting of the parameter ω, either diversity and coverage or convergence and fast computational time would be the favoured features when executing HVEA.

The chapter also extensively studies the multiple 0/1 knapsack problem using different greedy repair methods. In order to assess the performance of EMO algorithms fairly, the same greedy repair method should be used. Furthermore, this study suggests that the greedy repair method proposed by Valenzuela [115] should be deployed while assessing the performance of EMO algorithms on the multiple 0/1 knapsack problems in order to minimise the effect of the greedy repair method on the overall performance.

With regard to further studies, it is of interest to investigate the incorporation of the steady-state selection, deployed by SEAMO2 and MOEA/D, into fitness and/or ranking assignment population-based EMO algorithms. The reason for this proposals is that steady-state selection allows strong offspring to immediately participate in the recombination process of the current generation and this could lead to better convergence whereas the fitness and/or ranking assignment population-based EMO algorithms consider the whole population during the environmental selection which could lead to a better diversity.

Chapter 7

Theoretical Model To Real-world Nurse Scheduling Problems

7.1 Introduction

This chapter investigates the possibility of applying well-developed EMO algorithms to solve nurse scheduling problems. An introduction to real-world nurse scheduling was given in chapter 2. That chapter also discussed several evolutionary algorithm approaches to tackle this type of problems and it is observed that there is a great variation on how such algorithms operate on this problem. These approaches are normally based on a simple evolutionary algorithm incorporating problem-specific heuristics/metaheuristics. These heuristics/metaheuristics are usually designed to tackle a specific nurse scheduling problem with a given set of constraints. Then such approaches could be extended to a class of problems with a similar set of constraints. Chapter 3 presented a brief review of several general EMO algorithms from the literature. These well-developed EMO algorithms show strong performance on several benchmark problems. Many of these benchmark problems have a smaller number of constraints and are easier to formulate (com-

pared to nurse scheduling) although still present huge search spaces. These properties allow simple heuristics to construct feasible solutions for a given problem. The problem is then optimised by EMO algorithms. Due to the moderate effect on the search results of simple heuristics (compared to the effect of the EMO algorithms), it is relatively easy to compare and assess the performance of different EMO algorithms on such benchmark problems. Despite the strong performance of EMO algorithms, there is not much work in the literature on the application of general EMO algorithms to solve real-world nurse scheduling problems. The reason behind this is that real-world nurse scheduling problems are normally highly constrained and problem-specific heuristics/metaheuristics are usually required to generate feasible solutions. These heuristics are often elaborate techniques which could highly influence the overall search results. The effect of these problem-specific heuristics is likely to override the effect of general EMO algorithms on the search result. This is one of the main issues to address in the deployment of general EMO algorithms to solve real-world nurse scheduling problems. This chapter will address this issue. The main purpose of this chapter is to attempt to bridge the gap between the field of general EMO algorithms and the field of real-world nurse scheduling problems. It will show that general EMO algorithms themselves could solve real-world nurse scheduling problems without requirement of any problem-specific heuristics. Such results obtained by general EMO algorithms could be considered as the base-line performance against which other problem-specific heuristics could be compared.

The nurse scheduling problem in the Ophthalmological ward at the Queens Medical Centre University Hospital NHS Trust (QMC) in Nottingham UK, which was originally described by Beddoe [7], is used in this investigation. The QMC problem has been discussed by several researchers using different approaches which include case-based reasoning [103, 7, 8, 9, 10] and evolutionary multiobjective algorithms [81]. It is noted that different approaches used slightly different models for the QMC problem. However the data sets for the QMC problem are exactly the same and the set of constraints together with their associated parameters are

highly similar. The QMC problem as described by Landa-Silva and Le [81] is further investigated in this chapter. The reasons for using the QMC problem as described in [81] are as follows:

- The QMC problem covers a wide set of the most common constraints in nurse scheduling literature as identified in [30] together with their reasonable associated parameters which could be likely encountered in other real-world nurse scheduling problems.
- The QMC model developed by Landa-Silva and Le [81] is more suitable to the framework of EMO because the model itself was developed to be optimised by a simple evolutionary algorithm for multiobjective optimisation (SEAMOR).
- This QMC model clearly defined a set of hard constraints and a set of soft constraints which are quite balanced in term of the size of each set. It makes the problem neither too easy nor too hard to construct feasible solutions.
- Finally, the 7 data sets of the QMC problem vary significantly regarding the available resources (available staff-hours). According to Landa-Silva and Le [81], in term of available staff-hours, one data set is just above the minimum requirement, two data sets are slightly under the minimum requirement, two data sets are quite under the minimum requirement and two data sets are heavily under the minimum requirement. This characteristic of the 7 data sets not only makes the QMC problem challenging to be solved but also represents considerable variability in the instances difficulty which helps to assess how effective general EMO algorithms are in solving the QMC problem.

Chapter 3 reviewed a number of general EMO algorithms in the literature. Chapter 3 also discussed the multiple 0/1 knapsack problem which is widely accepted as a benchmark problem to assess the performance of EMO algorithms. The four EMO algorithms, which are under investigation in this chapter are

NSGA2 [47], SPEA2 [124], SEAMO2 [96] and HVEA (proposed in chapter 6) which show strong performance on the multiple 0/1 knapsack problem. There are several representations and their associated greedy repair methods for the multiple 0/1 knapsack problem. Different representation and greedy repair method tend to give different effect on the performance of EMO algorithms. In order to minimise this effect, it is suggested to use the permutation representation with a decoder proposed by Valenzuela [115] (as in section 6.4). The decoder is a very simple heuristic which accepts an item included in the knapsacks if the knapsack capacity constraint is not violated. This chapter will show how to model the QMC problem using the multiple 0/1 knapsack problem by extending this simple principle of solution decoder.

7.2 Queens Medical Centre Problem

The ophthalmological ward in QMC consists of about 30 nurses. Depending on data sets, there are around 20 available nurses including both part-time and full-time nurses. Schedules are produced for 28 day periods and cover is required on a 24 hour basis, 7 days a week. According to Beddoe [7], schedules at the QMC ward are manually constructed as a three stage process:

1. Based on the skills, nurses are grouped into teams.
2. Preference schedules for the planning period are produced by individuals in consultation with other team members.
3. The ward schedule is produced by combining individual preference schedules. Head nurses, then, repair constraint violations in the ward schedule.

When repairing the ward schedule, head nurses aim to maintain individual preference schedules as much as possible.

7.2.1 Problem Description

As described by Beddoe [7], in the QMC problem, each nurse works either on a full-time (FT) or part-time (PT) basis. Nurses are classified in a hierarchy according to their qualifications. There are four possible qualification categories: *registered* (RN), *enrolled* (EN), *auxiliary* (AN) and *student* (SN). Registered nurses are the most qualified and have received extensive training in both practical and managerial aspects of nursing, whereas enrolled nurses have received mainly practical training. Registered and enrolled nurses are classified as *qualified* (QN). Auxiliary nurses are unqualified nurses and student nurses are training to be either registered or enrolled. Qualified and auxiliary nurses are both *employed* (PN). Qualified nurses can receive additional training according to the ward that they work in. In the ophthalmological ward, these nurses receive *eye-training* (ET).

In the QMC problem, there are three types of normal working shifts: *early* (E), *late* (L), *night* (N). The early shift is from 07:00 to 14:45 counting for seven and a half hours (7.5 hours). The late shift is from 13:00 to 21:15 counting for seven and a half hours (7.5 hours). The night shift is from 21:00 to 07:15 counting for ten hours (10 hours). Occasionally, nurses indicate in their individual preference schedules the starting and finishing time that they prefer to work instead of one of the above normal shifts. In that case, the unusual shift is considered as the normal shift (early, late or night shift) that covers most of the hours of the unusual shift. For example, an unusual shift from 09:00 to 17:00 is considered as an early shift. If the unusual shift is equally spread over two adjacent normal shifts, one of these normal shifts is uniformly chosen at random. For example, an unusual shift from 17:00 to 01:15 can be considered as a late or as a night shift.

In the QMC problem, for each shift, there is a different coverage demand, i.e. the required number of nurses with specific qualifications and training. For each scheduling period, nurses could specify their individual working preferences (e.g.

days off, preferred shifts, etc.). There are also a number of working regulations, including nurses *annual leave* (AL), which must be followed. Then, the problem is to construct a schedule that meets the workforce demand, satisfies all regulations and meets as many individual preferences as possible. Landa-Silva and Le [81] defined the sets of hard and soft constraints as follows:

Hard Constraints

OneShiftADay A nurse works at most one shift (early, late, or night) each day.

MaxHours The maximum number of hours, which a nurse N_i could work, is defined by $Hours_i$ over a period of time according to each individual contract.

MaxDaysOn Every nurse only allows to work the maximum of 6 consecutive days. This constraint guarantees regular breaks for nurses.

MinDaysOn Full-time nurse must work at least 2 day consecutively. This constraint is not applicable for most part time nurses because of the fewer number of shifts that they work.

Succession There are certain working shift patterns which must not be scheduled, i.e. a night shift must not be followed by an early shift.

HardRequest Occasionally, nurses request annual leaves in their individual preference schedules. These requests must be satisfied at all time.

Soft Constraints

SoftRequest Apart from annual leave, nurses could indicate working shifts or day-offs which they prefer for any particular days. These are desirable but might be violated.

SingleNight Nurses at the QMC ward prefer to work night shifts in blocks of two or more. This applies to all full time nurses and certain types of part time nurses whose individual contracts are at least 20 hours a week.

WeekendSplit Nurses prefer to work both days of the weekend or none at all.

WeekendBalance The maximum number of weekends that nurses may work over the scheduling period. In the QMC ward, nurses may not work more than 3 out of 4 consecutive weekends.

Coverage A certain number of nurses with specific qualifications and specific training should be assigned to particular shifts as shown in Table 7.1.

Table 7.1: *Coverage* demand of nurses in each shift.

	Early	Late	Night
QNs	4	3	2
RNs	1	1	0
ETs	1	1	1

7.2.2 Problem Modelling

Landa-Silva and Le [81] formulate the QMC problem as an ordered pair:

$$NRP = \langle Nurses, C \rangle \tag{7.1}$$

where $Nurses = \{N_i : 1 \leq i \leq n\}$ is a set of n nurses and C is a set of constraints. Constraints in C can be hard (must be satisfied) or soft (should be satisfied). A nurse N_i is defined as follows:

$$N_i = \langle Detail_i, Preference_i, Schedule_i, GeneSeq_i \rangle \tag{7.2}$$

$Detail_i = \langle Contract_i, Qualification_i, Trained_i, Hours_i \rangle$

$Contract_i \in \{FT, PT\}$ nurse N_i is either full-time or part-time.

$Qualification_i \in \{RN, EN, AN, SN\}$ nurse N_i belongs to one of four qualification categories.

$Trained_i \in \{NoTrained, Trained\}$ in the ophthalmological ward, qualified nurses can receive eye-training.

$Hours_i \in N^+$ is the number of contracted hours for nurse N_i, for full-time nurses $Hours_i$ is 75 hours per fortnight, for part-time nurses $Hours_i$ is given per week as specified in their individual contract.

$Preference_i = \{p_{i,j} : 1 \leq j \leq NoOfDays\}$ is the individual preference schedule of nurse N_i for the scheduling period, where $NoOfDays$ is the length of the scheduling period, 28 in the QMC problem, and $p_{i,j} \in \{AL, E, L, N, O, U\}$ is the preference of nurse N_i for day jth of the scheduling period. O indicates a day-off and U indicates no specific preference. In figure 7.1, the individual preference schedule illustrates $Preference_i$.

$Schedule_i = \{s_{i,j} : 1 \leq j \leq NoOfDays\}$ is the individual schedule for nurse N_i, and $s_{i,j} \in \{AL, E, L, N, O, U\}$ is the assigned shift for nurse N_i on day jth of the scheduling period (U indicates unassigned day). Then, a ward schedule for the QMC problem is a collection of n individual nurse schedules. In figure 7.1, the constructed individual schedule illustrates $Schedule_i$.

$GeneSeq_i =$ Permutation $\{ensh : 1 \leq ensh \leq NoOfDays * NoOfShift\}$ is the encoded individual schedule for nurse N_i which is used in the EA framework. It is noted that this encoded schedule only represents the working shift patterns for nurse N_i, i.e. $NoOfShift = 3$ for early (E), late (L) and night (N) shift. It does not include annual leaves (AL) and day-offs (O). In figure 7.1, the permutation list of shifts illustrates $GeneSeq_i$.

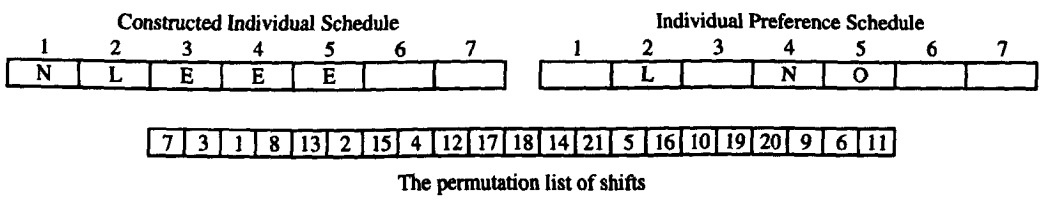


Figure 7.1: Constructed Schedule, Preference Schedule and Permutation List

In this model, hard constraints must be satisfied and this is enforced in the constructed schedules. However, soft constraint violations are accepted but penalised. The quality of a feasible schedule is then assessed based on these penalties on soft constraint violations. Landa-Silva and Le simply count the number of violations of each soft constraint type. In order to serve the principle of EMO approach, the set of soft constraints are split into four groups which correspond to four objective functions:

- Objective function 1 measures the level of nurse preferences satisfaction (i.e. *SoftRequest* constraint).
- Objective function 2 measures satisfaction of work regulations (i.e. *SingleNight*, *WeekendBalance* and *WeekendSplit* constraints).
- Objective function 3 measures the shortfalls in workforce demand (i.e. *Coverage* constraints).
- Objective function 4 measures the distribution of nurses in the schedule to ensure a balanced coverage for the whole scheduling period.

It is noted that the objective function 4 is to deal with an additional soft constraint which is proposed by Landa-Silva and Le to the original QMC problem introduced by Beddoe [7]. This additional soft constraint attempts to evenly distribute the number of nurses assigned to each shift over the scheduling period. Any surplus/shortage of nurses over the scheduling period should be kept to a minimum. This constraint prevents an excessive number of nurses being assigned to a particular shift while having a shortage of nurses in other shifts. The objective function 4 measures the satisfaction of this constraint based on statistical variation on the difference between the number of qualified nurses assigned to each shift and the coverage demand for qualified nurses. For example, for a schedule of 1-day and 3 shifts (Early, Late, Night), the difference between coverage demand and assigned qualified nurses is calculated as in table7.2. Then, the objective function 4

is measured as the variation on the Difference set of $3 * NoOfDays$ shifts. In this example, the value of the objective function 4 is $\frac{(0-0)^2+(1-0)^2+(-1-0)^2}{3} = \frac{2}{3} = 0.667$.

Table 7.2: Measurement of the distribution of nurses (the objective function 4).

Qualified Nurses	Early	Late	Night
Demand	4	3	2
Assigned	4	4	1
Difference	0	1	-1

Landa-Silva and Le also pointed out that due to the shortfall of staff in hospitals recently nurses satisfaction is at the centre of the scheduling process in the QMC problem. Therefore, they pre-set a target value for objective function 1 to guarantee a minimum level of staff satisfaction while constructing schedules. The other 3 objective functions are subject to optimisation by SEAMOR, an algorithm based on the framework of SEAMO proposed by Mumford (Valenzuela) [96, 115] with a re-generation strategy. The re-generation strategy was designed to activate the production of good-quality offspring to tackle the stagnation issue encountered by SEAMO when applied to the QMC problem. See [81] for more details on the measurement objective functions.

7.2.3 Self-Adaptive Heuristic Decoder

Landa-Silva and Le implemented a self-adaptive decoder for their QMC model. Based on the individual encoded schedules ($GeneSeq_i$), the self-adaptive decoder constructs individual schedules which are combined to produce the ward schedule for the planning period. The decoder repeatedly chooses a nurse N_i at random to create the individual schedule by decoding the encoded individual schedule $GeneSeq_i$ for that nurse N_i until all individual schedules are constructed to form the ward schedule. For each nurse N_i, the decoder starts from the beginning of the encoded individual schedule $GeneSeq_i$ and then assigns shifts to days until the end of the encoded individual schedule $GeneSeq_i$ is reached. For each encoded shift *ensh* in the $GeneSeq_i$, the corresponding working shift *shift* and assigned day j

are as follows:

$$j = \left\lfloor \frac{ensh - 1}{NoOfShift} \right\rfloor + 1 \tag{7.3}$$

$$sh = ensh - (j - 1) * NoOfShift \tag{7.4}$$

sh is 1, 2, or 3 corresponding to *early* (E), *late* (L) or *night* (N) shift respectively. For example (figure 7.2), an encoded shift $ensh = 13$ implies day $j = 5$ and $sh = 1$ (or $shift = E$) which means the *early* shift is assigned to 5th day of the planning period for nurse N_i ($s_{i,5} = E$). This is the provisional assignment, which is accepted if it does not violate any hard constraints.

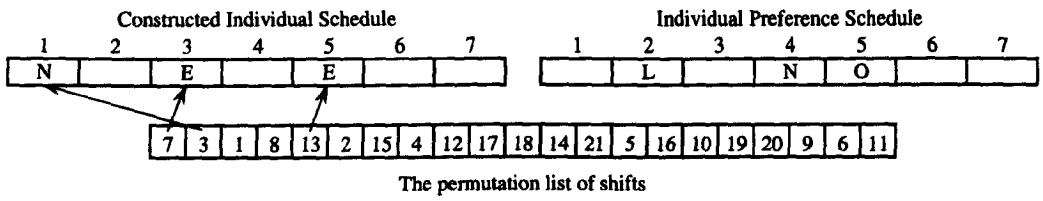


Figure 7.2: The decoding process.

However, if the provisional assignment $s_{i,j} = shift$ violates hard constraints, the provisional assignment is either rejected or repaired. The provisional assignment is rejected if it violates any of these hard constraints, *HardRequest*, *OneShiftADay*, *MaxHours* or *MaxDaysOn*. For example (figure 7.2), the decoder starts from the beginning of the permutation list $GeneSeq_i$ for nurse N_i and assigns shifts to days as follows: $s_{i,3} = E$ ($ensh = 7$), $s_{i,1} = N$ ($ensh = 3$). However, the next two encoded shifts $ensh = 1$ and $ensh = 8$, corresponding to $s_{i,1} = E$ and $s_{i,3} = L$ respectively, violate the *OneShiftADay* hard constraint hence these two assignments are rejected. The next encoded shift $ensh = 13$ leads to the assignment $s_{i,5} = E$. If *Succession* hard constraint is violated, the provisional assignment is repaired by allocating a different shift to day j. For example (figure 7.3), decoding $ensh = 4$ corresponding to assigning $s_{i,2} = E$ violates *Succession* hard constraint as it creates an illegal pattern (Night-Early). Therefore this provisional assignment is repaired

by assigning $s_{i,2} = L$ from the individual preference schedule. If *MinDaysOn* hard constraint is violated, an additional shift is assigned to one of the days adjacent to day j (figure 7.4).

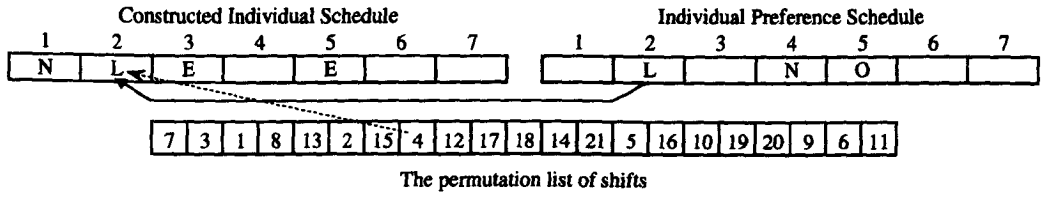


Figure 7.3: Repair the violation of the *Succession* constraint.

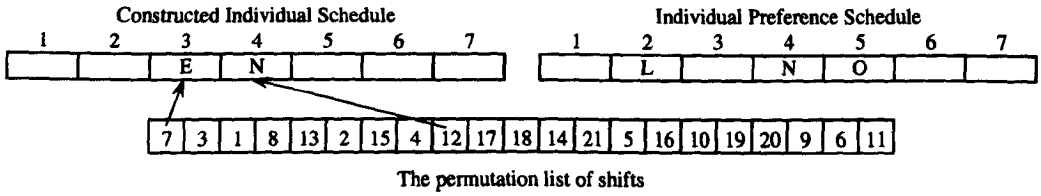


Figure 7.4: Repair the violation of the *MinDaysOn* constraint.

Besides repairing the *MinDaysOn* and *Succession* hard constraints, the decoder also attempts to minimise the number of surplus nurses in each shift. This is to reduce the number of violations of the *Coverage* constraint and equally distribute nurses amongst all shifts. This is achieved by only assigning shifts to days if the coverage demand has not been exceeded yet. The main purpose of this strategy is to deal with the shortage of available staff-hours in the QMC problem. Furthermore, the decoder also mimics the process of the real manual schedule construction by head nurses. As in the manual process, the decoder always attempts to satisfy individual preferences while repairing constraint violations (as shown in figure 7.3. An individual preference is accepted at a pre-defined probability if there is no violation of hard constraints. The purpose of this approach is to construct schedules which meet the pre-set level for individual preference satisfaction measured by objective function 1. See [81] for more detail on the evolutionary algorithm approach for the QMC problem.

The SEAMOR produces a good set of alternative solutions to the QMC problem [81]. However it is quite elaborate and could be difficult to apply to other

problems or even the same problem with different hard constraints. Section 7.3 examines a more general heuristic approach based on the principle of the heuristic approach employed by Mumford (Valenzuela) for the multiple 0/1 knapsack problem. The QMC model proposed by Landa-Silva and Le [81] is used to assess the performance of the new proposed approach.

7.2.4 Preliminary Investigations

This section discusses the model and the heuristic approach for the QMC problem in [81]. Based on these discussion, new models and simpler heuristic approaches are examined.

In the QMC model proposed by Landa-Silva and Le [81], one of the objectives measures the shortfall in the coverage demand and another objective measures the distribution of nurses for each shift over the planning period. Initial experimental results show that these two objectives are highly correlated with a value of around 0.9. The reason could be that the heuristic only allows shift assignments that do not exceed the coverage demand. Therefore, it is reasonable to only consider the objective that measures violations on the coverage demand. In order to maintain the 3-dimensional search space, the objective function that measures the individual preferences satisfaction is optimised rather than being pre-set as a target.

In [81], the decoder deployed a heuristic to repair the *MinDaysOn* hard constraint. During the decoding process, this heuristic assigns a shift to the adjacent day of day jth of the provisional assignment. It is noted that, at the beginning of the decoding process, there could be a high number of violations on the *MinDaysOn* hard constraint due to the partially filled nurse schedules. However, towards the end of the decoding process while the nurse schedule is filled by assignments, violations on the *MinDaysOn* hard constraint could be much lower, or eventually eliminated. Therefore, if the *MinDaysOn* constraint is considered when

the decoding process finishes, it might be not necessary to deploy this repairing heuristic.

The initial investigation presented here also examines the performance of different EMO algorithms (NSGA2, SPEA, SEAMO2 and HVEA) on solving the QMC problem using the heuristic decoder proposed by Landa-Silva and Le [81]. Figure 7.5 presents the search progress of these algorithms, measured using the hypervolume metric. In Figures 7.5(a)-7.5(h), the vertical axes indicate the percentage of the dominated hypervolume and the horizontal axis indicates the hypervolume value achieved throughout the search progress which is based on one million evaluations using different population size of 50 (Figure 7.5(a), 7.5(c), 7.5(e), 7.5(g)) and 200 (Figure 7.5(b), 7.5(d), 7.5(f), 7.5(h)). Figure 7.5 only shows results on 4 data sets which are representative of the 7 data sets. Due to the similarity between June2001 and July2001; April2001 and August2001; May2001 May2001 and September2001 in terms of the available nurse-hours and results, data obtained for instances July2001, August2001 and September2001 is not shown but algorithm assessment is based on all 7 data sets.

Figure 7.5(a), 7.5(c), 7.5(e) and 7.5(g) shows that when using small population size of 50 individuals the results are similar for all algorithms. The reason behind this is that solutions produced by the heuristic decoder are already quite good. It is difficult for EMO algorithms to further improve the solutions set especially when using a small population of size 50. In other words, the heuristic decoder highly influences the search performance when using a small population. Figure 7.5(b), 7.5(d), 7.5(f) and 7.5(h) shows the performance of the heuristic decoder in EMO algorithms by using a larger population size of 200 individuals. When using a large population, there are more solutions in the population some of them may not yet be good enough. Therefore EMO algorithm could further improve the solutions set slightly. The effect of the heuristic decoder seems to be reduced slightly but still considerable especially when being incorporated into NSGA2, SPEA2 and HVEA.

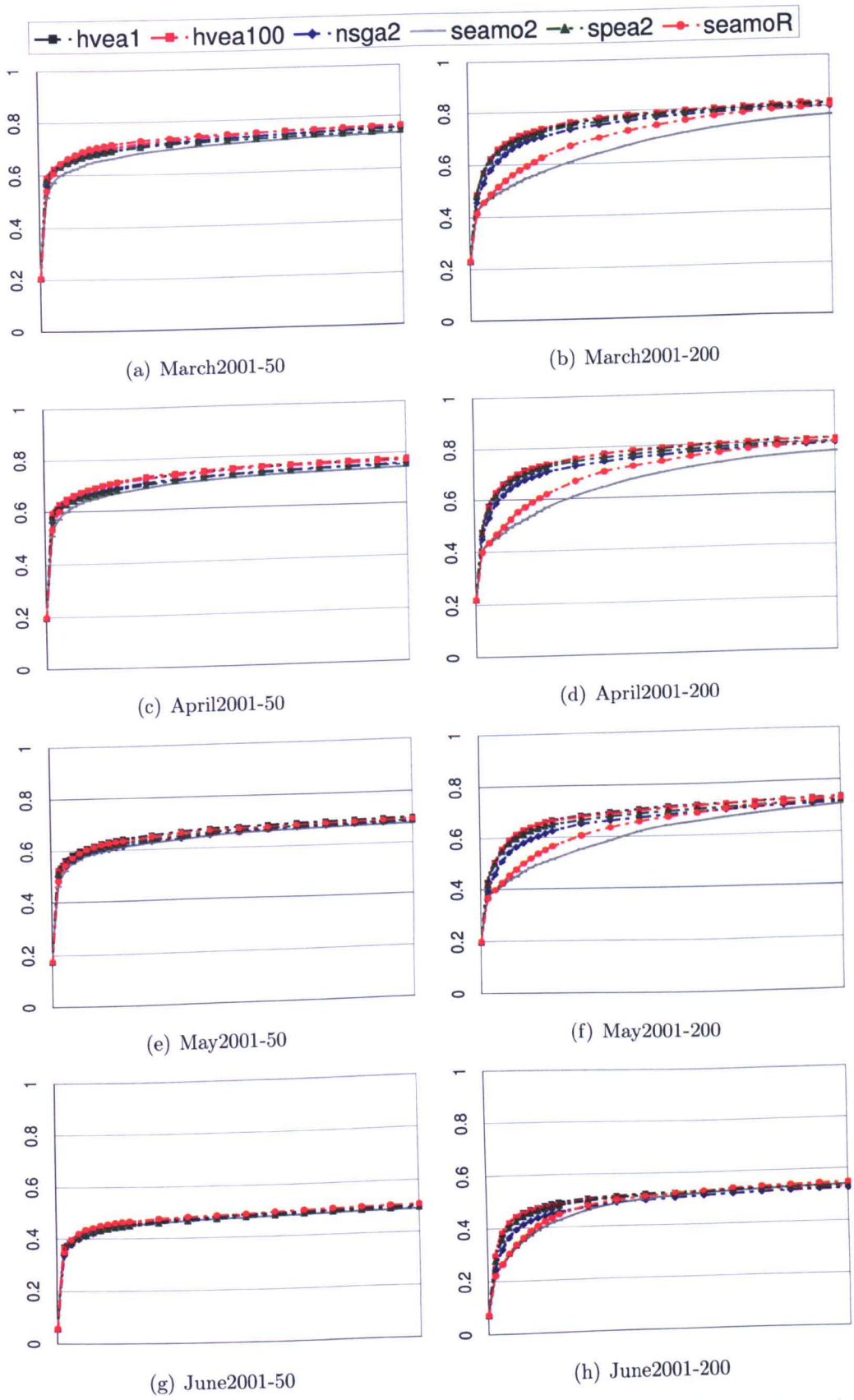


Figure 7.5: Performance, based on the $\mathcal{S}$-metric, of EMO algorithms incorporated the heuristic decoder on the QMC model proposed in [81].

7.3 A New Model for QMC Problem

This section discusses a new model for the QMC problem described by Landa-Silva and Le [81] together with a simple heuristic decoder. In the new QMC model, one single permutation represents the encoded ward schedule whereas the previous model represents the ward schedule as n individual encoded nurse schedules. Three objective functions, which are optimised in the new QMC model, address three criteria: the satisfaction of work regulations, the shortfall in coverage demand of shifts over the planning period and the individual preference satisfaction. The heuristics decoder for the new QMC model is very simple. By decoding the encoded shift in the permutation, a shift assignment for a nurse is obtained. The decoder only accepts this shift assignment if it does not violate any of these hard constraints: *OneShiftADay*, *MaxHours*, *MaxDaysOn*, *Succession*, *HardRequest*. If there is any violation on *MinDaysOn*, which makes the solution infeasible, this solution is heavily penalised by multiplying its objective functions $\alpha = 100$ times so that this infeasible solution is identifiable by optimisers. Then this new proposed decoder is incorporated into EMO algorithms to solve the QMC problem.

This simple heuristic decoder was actually developed based on the principle of the decoder proposed by Mumford (Valenzuela) [115, 96] for the multiple 0/1 knapsack problem which terminates the packing process if the knapsack capacity is exceeded. In other words, the knapsack item is not accepted if the hard constraint is violated. With an analogy to this principle, in the new QMC model, if a shift assignment violates hard constraints, the shift assignment is not accepted. However, the decoding process continue until the end of the permutation rather than terminating. A relaxed condition is applied to *MinDaysOn* hard constraint which heavily penalises infeasible solutions that violate this hard constraint.

The new QMC model had been constructed based on further investigation on the model proposed in [81]. The rest of this section presents this investigation.

7.3.1 Hard Constraint Violation Repaired Heuristics

The first attempt to reduce the effect of the heuristic decoder is based on the suggestion aforementioned in section 7.2.4. The objective function 1 that measures the individual preferences satisfaction is optimised instead of the objective function 4 that measures the distribution of nurses. The heuristic repairing the violations of the *MinDaysOn* hard constraint is also opted out in the new approach. This could lead to infeasible solutions which are heavily penalised in order to be identified as bad solutions by optimisers. Each objective of these infeasible solutions are multiplied by $\alpha = 100$. Another component of the heuristic decoder, which is the repair for the *Succession* hard constraint violation, is also opted out in the approach proposed here. Rather than finding a different assignment to repair the *Succession* constraint violation, the same approach used to deal with other hard constraints such as *MaxHours* or *MaxDaysOn* is employed. This means that the provisional assignment is not repaired, but rejected, when the *Succession* constraint is violated. Results are presented in Figure 7.6.

Figure 7.6 is presented in a format similar to that of figure 7.5. The experimental setting used for these experiments was also the same. In this new model of the QMC problem, the objective function 1 measuring the individual preferences satisfaction is optimised instead of the objective function 4 measuring the distribution of nurses. Even though, this objective function 4 is not optimised throughout the search, its value is reported and used to produce the results in figure 7.6 so that they could be compared to the results obtained by the model proposed by Landa-Silva and Le [81]. In Figure 7.6, SEAMOR are the results obtained from the QMC model and the heuristic decoder proposed in [81], i.e. SEAMOR in Figure 7.6 is the same as SEAMOR in Figure 7.5. Then, it is compared to HVEA, NSGA2, SPEA2 and SEAMO2 incorporating the heuristic decoder without repairing *MinDaysOn* and *Succession* hard constraint and using the mentioned objective function replacement. It is noted that only feasible solutions contribute to the re-

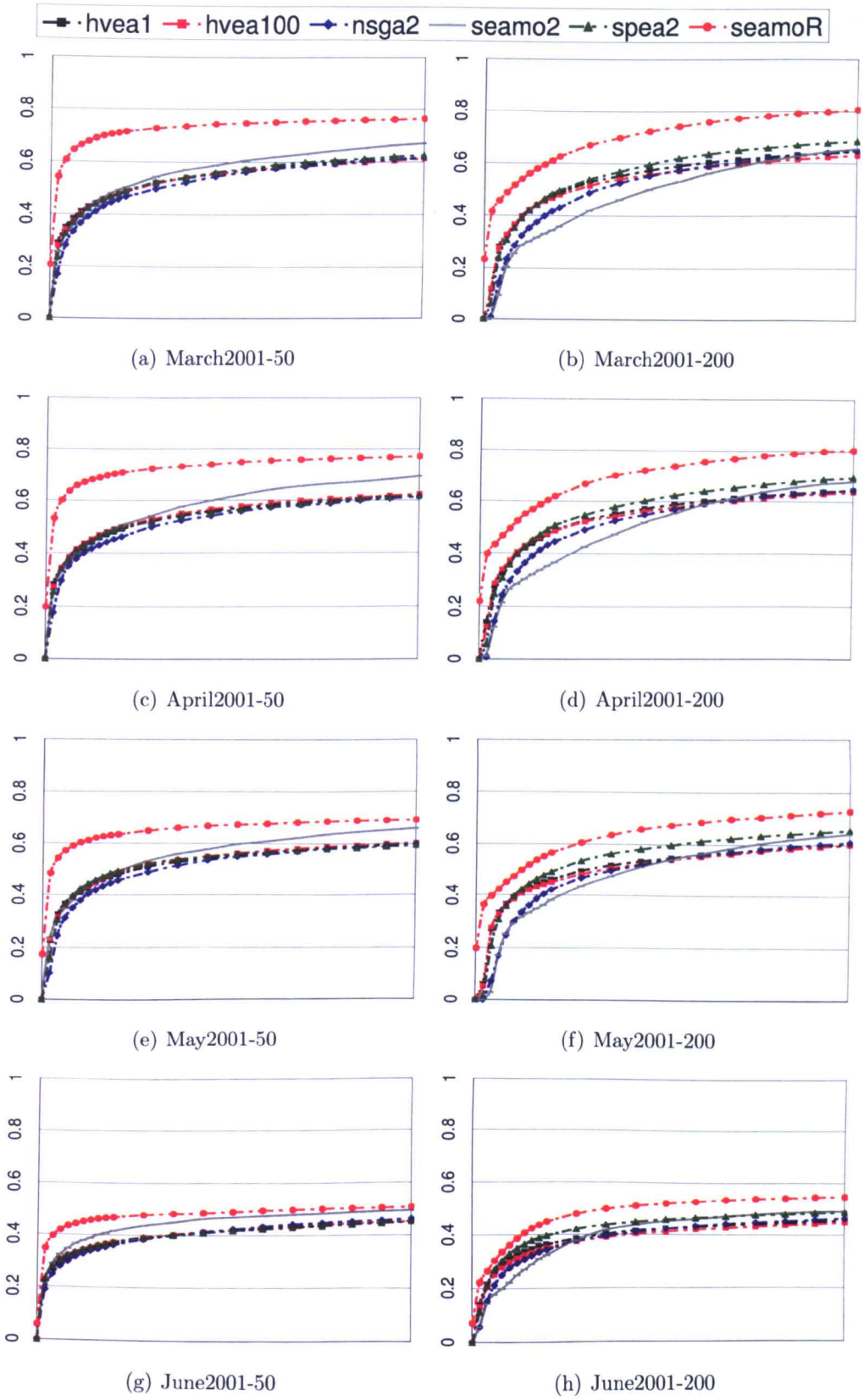


Figure 7.6: Performance, based on the $\mathcal{S}$-metric, of EMO algorithms incorporated the heuristic decoder without repairing *MinDaysOn* hard constraint on the QMC model proposed in [81] using objective function replacement.

sults presented in Figure 7.6. That figure shows that results obtained by EMO algorithms on this new QMC model are worse than results obtained by SEAMOR on the previous QMC model. However, it is promising to observe that feasible solutions are achievable. Additionally, there is evidence of the reduced effect that the heuristic decoder appears to have on the search process. Firstly, there are differences in the performance amongst HVEA, NSGA2, SPEA2 and SEAMO2 when using a population of 200 individuals. Secondly, the effectiveness of the search (i.e. increase in hypervolume) does not increase dramatically at the beginning of the search then improve only slowly towards the end of the search as it was the case in the results of Figure 7.5. Furthermore, another promising result is that by optimising the objective function that measures individual preferences satisfaction, the satisfaction level increases about 5%-15% compared to that obtained by SEAMOR solving the QMC model proposed in [81].

7.3.2 A More Deterministic Heuristic Approach

In this section, both the heuristic decoder proposed by Landa-Silva and Le [81] and the heuristic decoder examined in the previous section are applied to construct a nurse schedule for the QMC problem. As described in section 7.2.3, each nurse N_i is chosen at random and the heuristic decoder constructs a schedule for that nurse. This could lead to the situation in which early chosen nurses are fully assigned whereas later chosen nurses are partially assigned. This of course might not be the case for data sets with considerable shortage of available nurse-hours. However, it is likely the case for data sets with just about enough number of available nurse-hours. Therefore, to overcome this issue and to make the heuristic more deterministic, the $GeneSeq_i$ for all nurses are combined into a single large $GeneSeq$. Within the EA framework, instead of applying crossover operators on $GeneSeq_i$ to produce individual nurse schedule, crossover operators are now applied on the single large $GeneSeq$ to produce the ward schedule for the planning period.

The heuristic decoder is then applied onto *GeneSeq*. For each encoded shift *ensh* in the *GeneSeq*, nurse N_i, day j and shift *shift* are obtained as follows:

$$i = \left\lfloor \frac{ensh - 1}{NoOfDay * NoOfShift} \right\rfloor + 1 \tag{7.5}$$

$$j = \left\lfloor \frac{ensh - (i-1) * NoOfDay * NoOfShift - 1}{NoOfShift} \right\rfloor + 1 \tag{7.6}$$

$$sh = ensh - (i-1) * NoOfDay * NoOfShift - (j-1) * NoOfShift \tag{7.7}$$

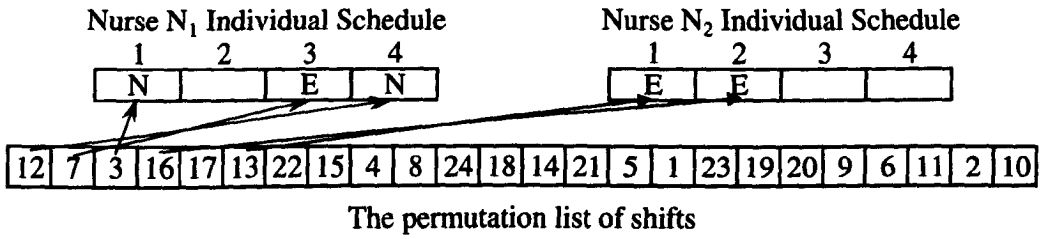
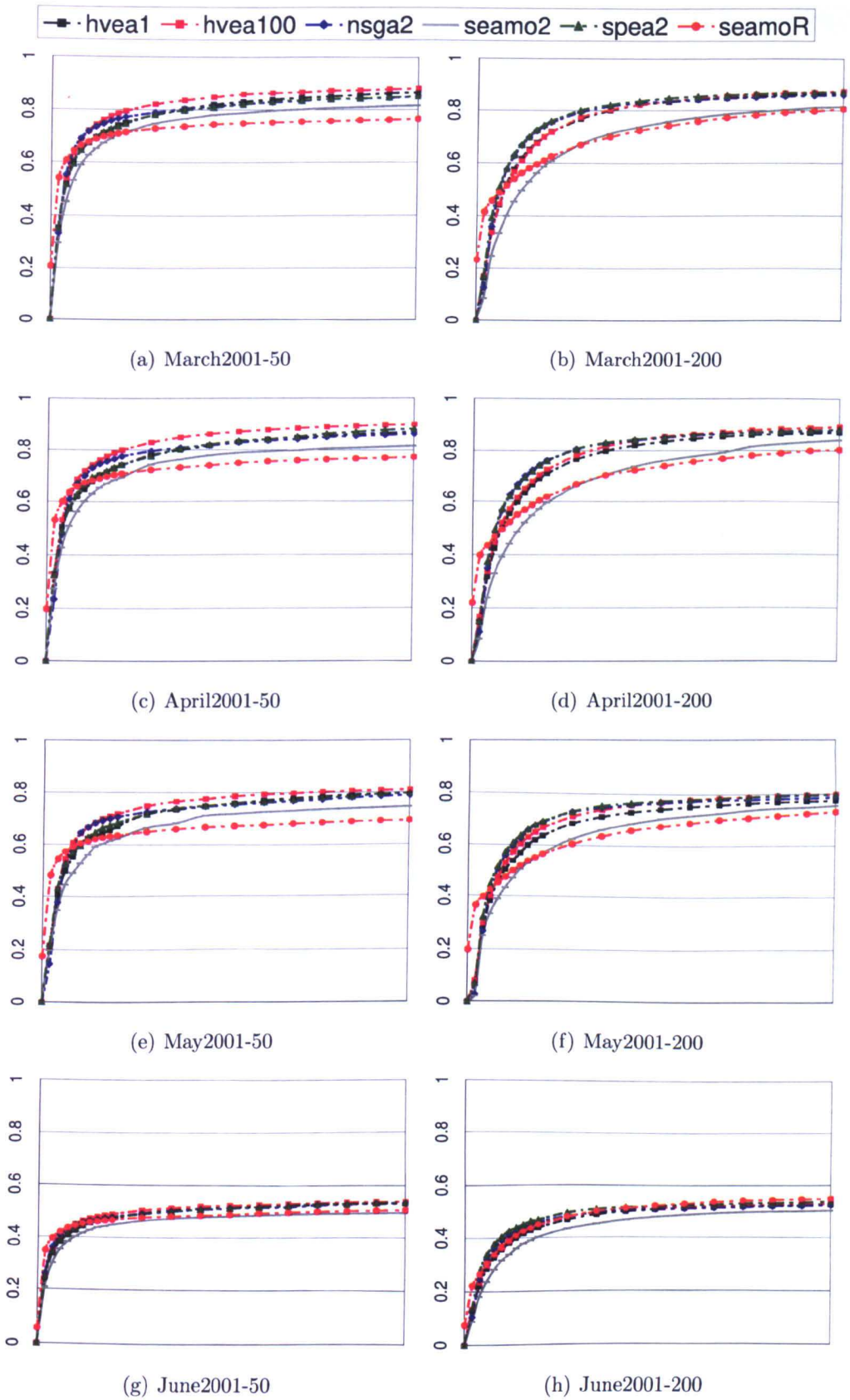


Figure 7.7: The decoding process using single permutation list of shifts.

sh takes the value of 1, 2 or 3 corresponding to *early* (E), *late* (L), *night* (N) shift respectively. Also to reduce the effect of the heuristic decoder, the proposal opts out the heuristic component which only allows the shift assignment if the coverage demand is not exceeded. The results of this new heuristic decoder and the new model for the QMC problem are presented in Figure 7.8. It shows that the new model for the QMC using this more deterministic decoder which is then optimised by NSGA2, SPEA2, SEAMO2, or HVEA obtains better results than the one proposed in [81] when compared using the same assessment criteria. The results are not improved only on the most difficult data sets (June2001 and July2001) where the available nurse-hours are well below the minimum requirement. Furthermore, the satisfaction level of nurses' preferences increases about 10%-17%, except in the results obtained by SEAMO2 (5%-10%) compared to that obtained by SEAMOR solving the QMC model proposed in [81]. It is noted that there are few extreme solutions obtained by the new model. These solutions have low values in one objective but high values in other objectives. However, by removing these extreme solutions from the non-dominated sets found, the performance are still very similar to that of non-dominated sets which include these extreme solutions.



(a) March2001-50

(b) March2001-200

(c) April2001-50

(d) April2001-200

(e) May2001-50

(f) May2001-200

(g) June2001-50

(h) June2001-200

Figure 7.8: Performance, based on the $\mathcal{S}$-metric, of EMO algorithms on the new QMC model using single permutation and a more deterministic heuristic decoder

7.4 A More General Approach to Nurse Scheduling Problems - A Proposal

This section proposes a general approach to nurse scheduling problems so that they can be solved by applying general purpose EMO algorithms. The nurse scheduling problems could be modelled based on the principle of the multiple 0/1 knapsack problem using the heuristic decoder proposed in [115, 96] as follows. Let's suppose that the knapsack capacity constraint is given by the upper bound for the multiple 0/1 knapsack problem. The heuristic decoder packs items into knapsacks upto this upper bound, then terminates. Therefore, let's assume that there exists the upper bound for the nurse scheduling problems. This upper bound instructs the heuristic decoder whether to accept shift assignments or not. The heuristic decoder moves onto the next shift assignment until the permutation is exhausted. The important issue is to identify a suitable upper bound for a given nurse scheduling problem. This upper bound could consist of a set of hard constraints or a hard constraint. The properties of these hard constraints are:

- Violations on these hard constraints are quickly and straightforwardly identifiable.
- It is not possible to repair violations on these hard constraints by any means. The only way is to not accept the shift assignment that generates these violations.

Any hard constraints which impose conditions on 'maximum values' could serve for the purpose of the upper bound. For examples: the maximum number of consecutive working days, the maximum number of consecutive working shifts, the maximum number of working hours over a period, etc.

Nurse scheduling problems are usually highly constrained and complex. One of the important issues when solving nurse scheduling problems is how to handle

complex constraints. Under the EMO framework, let's assume that soft constraints are grouped into objective functions to be optimised. However, it is also required to handle hard constraints to make solutions feasible. As mentioned early in this section, there is a type of violations on hard constraints which is not possible to repair, e.g. the maximum number of consecutive working days. To make solutions feasible, it is suggested to not accept the shift assignment that generates these violations. There is another type of violations on hard constraints which is possible to repair, e.g. the maximum number of consecutive day-offs. However, in order to keep the heuristics as simple as possible, these type of violations are accepted but heavily penalised so that infeasible solutions are identifiable by the optimisers. In other words, hard constraints associated to these violations are relaxed as soft constraints but violations on these constraints are heavily penalised. There is another type of violations on hard constraints which cannot be repaired by either accepting or rejecting the shift assignment, e.g. the maximum consecutive working days hard constraint is violated by accepting a shift assignment, but rejecting that shift assignment leads to an illegal shift pattern. For example, one weekend day-off is an illegal shift pattern (a nurse must either work both days during weekend or none), but assigning an additional shift to the weekend violates the maximum consecutive working days. If this is the case, the decision on either accepting or rejecting the shift assignment is based on satisfying the most important hard constraint or it could be randomly decided.

It is often the case that a given nurse scheduling problem consists of all soft constraints being satisfied as much as possible. The procedure to identify which constraints serve as the upper bound of the problem is based on the criteria proposed early in this section. Let's assume that a maximum of 6 consecutive working days is the soft constraint being served as the upper bound of the problem. However, this condition could not be used directly in the heuristic decoder to decide whether to accept the shift assignment. Instead, $(6 + \epsilon)$ consecutive working days could be used as an imaginary hard constraint that solutions must satisfy. This

value $(6 + \epsilon)$ could be considered as the maximum acceptance level of violation on this constraint. However this constraint is still treated as the soft constraint which need to be satisfied. Therefore, ϵ should be gradually reduced as the search progresses by assigning the average number of violations on this constraint to ϵ which gradually reduces ϵ. This is to make sure ϵ is also being optimised.

This general approach is drawn upon the investigation of the QMC problem. It has not yet been examined on other problems. Therefore, more investigation on the approach is required.

7.5 Summary

This chapter investigates the QMC model proposed by Landa-Silva and Le [81]. A new model and a much simpler heuristic decoder for general EMO algorithms to solve the QMC problem was investigated. It was shown that general EMO algorithms could be able to solve complex and highly constrained real-world nurse scheduling problems. This chapter also outlines a more general approach to nurse schedule problems to be solved by EMO algorithms. The approach is based on the investigation of the QMC problem, but it still needs to be developed and fully tested.

Chapter 8

Conclusions and Future Work

8.1 Conclusions

The work in the thesis presents an in-depth investigation on the EMO field and proposes a new form of relaxed Pareto dominance, namely *volume dominance*. Based on the concept of the *volume dominance*, a new EMO algorithm, *Hyper Volume Evolutionary Algorithm* (HVEA), is proposed. HVEA outperforms or remains competitive to various state-of-the-art EMO algorithms. The thesis also studies the application of general purpose EMO algorithms in solving the multiple 0/1 knapsack and a real-world nurse scheduling problems.

As discussed in the introduction of the thesis (Chapter 1), the EMO field has drawn much attention from researchers over the last decade or so. There is a number of strong performing EMO algorithms that have been proposed including NSGA2, SPEA2, SEAMO2, MOEA/D amongst others. However, relatively little work has been published on the application of these strong performing general EMO algorithms to solve real-world nurse scheduling problems. In this thesis, a general purpose EMO algorithm called HVEA is proposed. This thesis also outlines an approach for solving nurse scheduling problems using general EMO algorithms.

This approach is drawn upon the investigation on the QMC problem [81], a real-world nurse scheduling problem.

Within the field of EMO, one of the key issues in designing an EMO algorithm is to set a mechanism to establish superiority between solutions in the population. That is, mechanisms or criteria to discriminate between solutions in the multiobjective sense. However, this issue has not received much attention as it deserves. Most recent EMO algorithms deploy the conventional Pareto dominance relationship to compare solutions, but there are other types of dominance relationships, known as *relaxed* (or *extended*) *Pareto dominance*, in the literature. Chapter 5 presents review on a number of *relaxed Pareto dominance* such as α-dominance, ϵ-dominance, gain factor dominance, etc. Chapter 5 also proposes a new form of *relaxed Pareto dominance* called *volume dominance.* The improved version of *volume dominance* is also discussed in chapter 5. Extensive experiments are presented to compare the performance of both the initial proposal and the improved version of *volume dominance* against the conventional Pareto dominance using three EMO algorithms from the literature: SEAMO2, SPEA2 and NSGA2. Experiments were conducted on the multiple 0/1 knapsack problem and promising results were obtained by the improved *volume dominance.* The key observations were that this improved *volume dominance* is capable of obtaining a better and smoother trade-off front and is also more robust than Pareto dominance. To the best knowledge of the author, this chapter presents the first investigation on incorporating a *relaxed Pareto dominance* into different EMO algorithms.

Chapter 6 further explores the ideas of the *volume dominance* from chapter 5 and proposes a new population-based multiobjective evolutionary algorithm, the *Hyper Volume Evolutionary Algorithm* (HVEA). The concept of *volume dominance* is investigated and used as a new strategy to assign fitness to solutions in HVEA. In this new algorithm, the fitness of an individual is estimated based on the hypervolume of that individual without the requirement of determining the reference

point for the hypervolume calculation. HVEA also deploys a clustering technique which assesses the crowding of an individual by considering all individuals in its neighbourhood which is defined by the parameter ω, the *neighbouring area radius*. The HVEA approach outperforms or remains competitive to various state-of-the-art EMO algorithms including NSGS2, SEAMO2, SPEA2, IBEA$_{\epsilon+}$, IBEA$_{HV}$ and a very strong performance MOEA/D on the multiple 0/1 knapsack problem.

Chapter 6 also presents an extensive study on the multiple 0/1 knapsack problem using different greedy repair methods to fix the violations of capacity constraints. It is concluded that the same greedy repair method should be used for the multiple 0/1 knapsack problem to assess the performance of several EMO algorithms more fairly. It also suggested to use the greedy repair method proposed by Valenzuela [115] (to minimise the effect of the repair method) when assessing the performance of EMO algorithms on solving the multiple 0/1 knapsack problems.

Chapter 7 presents an investigation on the QMC model proposed by Landa-Silva and Le [81], a real-world nurse scheduling problem. The purpose of this study is to apply well-developed general EMO algorithms to solve real-world nurse scheduling problems. Based on the QMC problem, this study attempts to model real-world nurse scheduling problems with complex constraints as a multiple 0/1 knapsack problem. The study shows that special-purpose heuristics developed for nurse scheduling problems have strong effect on the search performance. Then, optimisers such as general EMO algorithms play a little role in the search performance if these special-purpose heuristics are incorporated. Chapter 7 also suggest a general approach to model real-world nurse scheduling problems in such a way that general EMO algorithms can be applied more effectively. It is suggested that results obtained by this approach could be used as the reference to assess the performance of special-purpose heuristics developed for a particular nurse scheduling problem. The study concludes that without special-purpose heuristics, it is possible for general EMO algorithms to solve complex and highly constrained real-world

nurse scheduling problems. To the best knowledge of the author, this study is the first investigation to bridge the gap between general EMO algorithms in the EMO field and nurse scheduling problems in the real-world.

Chapter 4, the final work of this thesis, studies restricted mating schemes in EMO algorithms. It proposes an adaptive assortative mating scheme that uses similarity in the decision space and adapts the mating pressure as the search progresses. Initial results show that constant values for the mating pressure in this scheme provoke either convergence or diversity to be negatively affected. The study suggests a simple adaptive scheme which varies σ_{mating} taking into account the population diversity in the decision space. Experiments show that this mechanism improve the performance of SEAMO2 [96] while striking a good balance between convergence and diversity.

8.2 Future Work

Based on the study came out of this thesis, these are four suggestions for future work:

1. It is noticed that in the literature, different forms of *relaxed Pareto dominance* have been proposed for different types of problems. It is interesting to see the performance of different *relaxed Pareto dominance* relationship while being incorporate into general EMO algorithms solving the same problems. Future work could to compare these alternative forms of dominance by incorporating them into SEAMO2 to solve the multiple 0/1 knapsack problem because SEAMO2 is very simple but yet effective. Other state-of-the-art EMO algorithms could be investigated later.

2. Durillo et al. [49] reported strong performance improvement by modifying NSGA2 and SPEA2 from generational to steady-state selection, but the

computational time increased significantly. This is due to the computation of individual fitness values when replacing a solution in the population by a better offspring. Other steady-state EMO algorithms such as SEAMO2, MOEA/D have also shown strong performance. Therefore, it is worthwhile to investigate the incorporation of steady-state selection in HVEA.

3. Other strategies to set the threshold mating pressure for the restricted mating scheme studied in chapter 4 to further improve diversity and convergence of EMO algorithms could be studied. Future work could also to investigate and compare different mating schemes within general EMO algorithms when applied to different problems such as real-world nurse scheduling problems. As it is known, besides demanding good values for the multiple objective functions, decision makers could be also interested in obtaining a diverse set of solutions in the decision space. This would provide a variety of alternative choices (different layouts for the schedules).

4. Chapter 7 outlined an initial study and proposal for a more general approach to modelling real-world nurse scheduling problems. This approach is drawn based on the investigation on the QMC problem. In order to clarify this approach, this approach needs to be tested on other nurse scheduling problems.

Bibliography

[1] W. Abernathy, N. Baloff, J. Hershey, and S. Wandel. A three-stage manpower planning and scheduling model: A service sector example. *Operations Research*, 22:693–711, 1973.

[2] Uwe Aickelin, Edmund Burke, and Jingpeng Li. An evolutionary squeaky wheel optimization approach to personnel scheduling. *IEEE Transactions on Evolutionary Computation*, 13:433–443, 2009.

[3] Uwe Aickelin and Kathryn Dowsland. Exploiting problem structure in a genetic algorithm approach to a nurse scheduling problem. *Journal of Scheduling*, 3:139–153, 2000.

[4] Uwe Aickelin and Kathryn Dowsland. An indirect genetic algorithm for a nurse scheduling problem. *Journal of Operations Research Society*, 31:761–778, 2003.

[5] Richard Allmendinger, Xiaodong Li, and Jürgen Branke. Reference point-based particle swarm optimization using a steady-state approach. In *Proceedings of the 2008 International Conference on Simulated Evolution and Learning*, volume 5361 of *Lecture Notes in Computer Science*. Springer, 2008.

[6] J. Arthur and A. Ravindran. A multiple objective nurse scheduling model. *AIIE Transactions*, 13:55–60, 1981.

[7] Gareth Beddoe. *Case-Based Reasoning in Personnel Rostering*. PhD thesis, School of Computer Science, University of Nottingham, 2004.

[8] Gareth Beddoe and Sanja Petrovic. Selecting and weighting features using a genetic algorithm in a case-based reasoning approach to personnel rostering. *European Journal of Operational Research*, 175(2):649–671, December 2006.

[9] Gareth Beddoe and Sanja Petrovic. Enhancing case-based reasoning for personnel rostering with selected tabu search concepts. *European Journal of Operational Research*, 58:1586–1598, 2007.

[10] Gareth Beddoe, Sanja Petrovic, and Jingpeng Li. A hybrid metaheuristic case-based reasoning system for nurse rostering. *Journal of Scheduling*, 12(2):99 – 119, 2009.

[11] Ilham Berrada, Jacques Ferland, and Philippe Michelon. A multi-objective approach to nurse scheduling with both hard and soft constraints. *Socio-Economic Planning Sciences*, 30(3):183–193, 1996.

[12] Nicola Beume, Boris Naujoks, and Michael Emmerich. Sms-emoa: Multi-objective selection based on dominated hypervolume. *European Journal of Operational Research*, 181:16531669, 2007.

[13] Tobias Blickle and Lothar Thiele. A comparison of selection schemes used in evolutionary algorithms. *Evolutionary Computation*, 4(4):361–394, 1997.

[14] Ivo Blöchliger. Modeling staff scheduling problems: A tutorial. *European Journal of Operational Research*, 158(3):533542, 2004.

[15] Peter Bosman and Edwin de Jong. Combining gradient techniques for numerical multi-objective evolutionary optimization. In *Proceedings of the 2006 Genetic and Evolutionary Computation*, volume 1, pages 627–634, 2006.

[16] Dimo Brockhoff and Eckart Zitzler. Improving hypervolume-based multi-objective evolutionary algorithms by using objective reduction methods. In *Proceedings of the 2007 Congress on Evolutionary Computation (CEC 2007)*, 2007.

[17] Peter Bruker, Edmund Burke, Timothy Curtois, Rong Qu, and Greet Vanden Berghe. A shift sequence based approach for nurse scheduling and a new benchmark dataset. *Journal of Heuristic*, 16(4):559–573, 2010.

[18] Edmund Burke, Peter Cowling, De Causmaecker Patrick, and Greet Vanden Berghe. A memetic approach to the nurse rostering problem. In *Special Issue: Simulated Evolution and Learning*, volume 15, pages 199–214. Springer, 2001.

[19] Edmund Burke, Timothy Curtois, Rong Qu, and Greet Vanden Berghe. A scatter search methodology for the nurse rostering problem. *Journal of Operational Research Society*, 61:Accepted for publication, 2010. Journal of Operational Research Society. Accepted for 2010 publication.

[20] Edmund Burke, Patrick De Causmaecker, Sanja Petrovic, and Greet Vanden Berghe. Fitness evaluation for nurse scheduling problems. In *Proceedings of the 2001 Congress on Evolutionary Computation*, volume 2, pages 1139 – 1146, 2001.

[21] Edmund Burke, Patrick De Causmaecker, Sanja Petrovic, and Greet Vanden Berghe. A multi criteria meta-heuristic approach to nurse rostering. In *Proceedings of the 2002 Congress on Evolutionary Computation*, volume 2, pages 1197 – 1202, 2002.

[22] Edmund Burke, Patrick De Causmaecker, Sanja Petrovic, and Greet Vanden Berghe. Metaheuristics for handling time interval coverage constraints in nurse scheduling. *Applied Artificial Intelligence*, 20(9):743–766, 2006.

[23] Edmund Burke, Patrick De Causmaecker, and Greet Vanden Berghe. Novel metaheuristic approaches to nurse rostering problems in belgian hospitals. In Joseph Leung, editor, *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, pages 44.1–44.18. CRC Press, 2004.

[24] Edmund Burke, Patrick De Causmaecker, Greet Vanden Berghe, and Hendrik Van Landeghem. The state of the art of nurse rostering. *Journal of Scheduling*, 7:441–449, 2004.

[25] Edmund Burke, Graham Kendall, and E. Soubeiga. A tabu-search hyperheuristic for timetabling and rostering. *Journal of H euristics*, 9:451–470, 2003.

[26] Edmund Burke and Dario Landa-Silva. The design of memetic algorithms for scheduling and timetabling problems. In *Recent Advances in Memetic Algorithms*, volume 166 of *Fuzziness and Soft Computing*, pages 289–311. Springer, 2005.

[27] Edmund Burke and Dario Landa-Silva. The influence of the fitness evaluation method on the performance of multiobjective search algorithms. *European Journal of Operational Research*, 169(3):875–897, 2006.

[28] Edmund Burke, Jingpeng Li, and Rong Qu. A pareto-based search methodology for multi-objective nurse scheduling. *Annals of Operations Research*, doi: 10.1007/s10479-009-0590-8, 2009.

[29] X. Cai and K.N. Li. A genetic algorithm for scheduling staff of mixed skills under multi-criteria. *European Journal of Operational Research*, 125:359–369., 2000.

[30] B. Cheang, H. Li, A. Lim, and B. Rodrigues. Nurse rostering problems: A bibliographic survey. *European Journal of Operational Research*, 151:447–460, 2003.

[31] Carlos Coello Coello. An updated survey of evolutionary multiobjective optimization techniques: State of the art and future trends. In *Proceedings of the 1999 Congress on Evolutionary Computation*, pages 3–13. IEEE, 1999.

[32] Carlos Coello Coello. A short tutorial on evolutionary multiobjective optimization. In *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, pages 21–40. Springer, 2001.

[33] Carlos Coello Coello, Gary Lamont, and David Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems.* Springer, 2007.

[34] Carlos Coello Coello, Gregorio Pulido, and Efren Montes. Current and future research trends in evolutionary multiobjective optimization. In *Information processing with Evolutionary Algorithms*, Advanced Information and Knowledge Processing, pages 213–231. Springer, 2005.

[35] Gualtiero Colombo and Christine Mumford. Comparing algorithms, representations and operators for the multi-objective knapsack problem. In *Proceedings of the 2005 Congress on Evolutionary Computation*, volume 2, pages 1268–1275, 2005.

[36] David Corne, Nick Jerram, and Martin Knowles, Joshua Oates. Peas-ii: Region-based selection in evolutionary multiobjective optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 283–290. Morgan Kaufmann, 2001.

[37] Charles Darwin. *The Origin of Species.* John Murray, 1859.

[38] Pallab Dasgupta, P. P. Chakrabarti, and S. C. DeSarkar. *Multiobjective Heuristic Search: An Introduction to Intelligent Search Methods for Multicriteria Optimization.* Computational Intelligence. Morgan Kaufmann, 1999.

[39] L. David. Applying algorithms to epistatic domains. In *Proceedings of the 1985 International Joint Conference on Artificial Intelligence*, pages 162–164, 1985.

[40] Susmita De, Sankar Pal, and Ashish Ghosh. Genotypic and phenotypic assortative mating in genetic algorithm. *Information Science*, 105:209–226, 1998.

[41] Kalyamoy Deb. Current trends in evolutionary multiobjective optimi. *International Journal for Simulation and Multidisciplinary Design Optimization*, 1(1):1–8, 2007.

[42] Kalyamoy Deb and David Goldberg. An investigation of niche and species formation in genetic function optimization. In *Proceedings of the Third International Conference on Genetic Algorithms*, pages 42–50, 1989.

[43] Kalyamoy Deb and Sundar. Reference point based multi-objective optimization using evolutionary algorithms. In *Proceedings of The 8th Annual Conference on Genetic and Evolutionary Computation*, pages 635–642, 2006.

[44] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms.* Wiley, 2001.

[45] Kalyanmoy Deb. Multi-objective optimization. In Edmund Burke and Graham Kendall, editors, *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, pages 273–316. Springer, 2005.

[46] Kalyanmoy Deb, Manikanth Mohan, and Shikhar Mishra. Towards a quick computation of well-spread pareto-optimal solutions. In *Proceedings of the 2003 International Conference on Evolutionary Multi-criterion Optimization*, volume 2623 of *Lecture Notes in Computer Science*, pages 222–236. Springer, 2003.

[47] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[48] Kalyanmoy Deb, J. Sundar, Udaya Bhaskara Rao N., and Shamik Chaudhuri. Reference point based multi-objective optimization using evolutionary algorithms. *International Journal of Computational Intelligence Research*, 2(3):273–286, 2006.

[49] Juan Durillo, Antonio Nebro, Francisco Luna, and Enrique Alba. On the effect of the steady-state selection scheme in multi-objective genetic algorithms. In *Proceedings of the 2009 International Conference on Evolutionary Multi-criterion Optimization*, volume 5467 of *Lecture Notes in Computer Science*, page 183197. Springer, 2009.

[50] A.T. Ernst, H. Jiang, M. Krishnamoorthy, B. Owens, and D. Sier. An annotated bibliography of personnel scheduling and rostering. *Annals of Operations Research*, 127:21–144, 2004.

[51] A.T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153:3–27, 2004.

[52] Carlos Fonseca. *Multiobjective Genetic Algorithms with Application to Control Engineering Problems.* PhD thesis, Department of Automatic Control and System Engineering, The University of Sheffield, 1995.

[53] Carlos Fonseca and Peter Flemming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In Forrest S., editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423. Morgan Kaufmann, 1993.

[54] L. S. Franz, H. M. Baker, G. K. Leong, and T. R. Rakes. Mathematical model for scheduling and staffing multiclinic health regions. *European Journal of Operational Research*, 41:277–289, 1989.

[55] David Goldberg. *Genetic Algorithms in Search, Optimisation and Machine Learning.* Addison Wesley, 1989.

[56] David Goldberg and Kalyamoy Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms.* Morgan Kaufmann, 1991.

[57] David Goldberg and Lingle R. Alleles, loci, and the traveling salesman problem. In *Proceedings of the International Conference on Genetic Algorithms and their Applications*, 1985.

[58] William Hart, Natalio Krasnogos, and Jim Smith. Memetic evolutionary algorithms. In *Recent Advances in Memetic Algorithms*, volume 166 of *Fuzziness and Soft Computing*, pages 3–27. Springer, 2005.

[59] William Hart, Natalio Krasnogos, and Jim Smith. *Recent Advances in Memetic Algorithms*, volume 166 of *Studies in Fuzziness and Soft Computing*. Springer, 2005.

[60] Kamrul Hasan, Ruhul Sarker, Daryl Essam, and David Cornforth. Memetic algorithms for solving job-shop scheduling problems. In *Memetic Computing*, volume 1, pages 69–83. Springer, 2008.

[61] Jeffery Horn, Nicholas Nafpliotis, and David Goldberg. A niched pareto genetic algorithm for multiobjective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation*, volume 1, pages 82–87. IEEE Press, 1994.

[62] Chien-Feng Huang. An analysis of mate selection in genetic algorithms. Technical report, Center for the Study of Complex Systems, University of Michgan, 2001.

[63] Kokoro Ikeda, Hajime Kita, and Shigenobu Kobayashi. Failure of pareto-based moeas: Dose non-dominated really mean near to optimal? In *Proceedings of the 2001 Congress on Evolutionary Computation (CEC 2001)*, volume 2, pages 957–962. IEEE Press, 2001.

[64] T. Inoue, T. Furuhashi, H. Maeda, and M. Takaba. A proposal of combined method of evolutionary algorithm and heuristics for nurse scheduling support system. *IEEE Transactions on Indus*, 50:5, 2003.

[65] Hisao Ishibuchi and Kaname Narukawa. Recombination of similar parents in emo algorithms. In *The 3rd International Conference on Evolutionary Multi-Criterion Optimization*, volume 3410 of *Leture Notes in Computer Science*, pages 265–279. Springer, 2005.

[66] Hisao Ishibuchi and Youhei Shibata. An empirical study on the effect of mating restriction on the search ability of emo algorithms. In *The 2nd International Conference on Evolutionary Multi-Criterion Optimization*, volume 2632 of *Lecture Notes in Computer Science*. Springer, 2003.

[67] Hisao Ishibuchi and Youhei Shibata. A similarity-based mating scheme for evolutionary multiobjective optimization. In *The 2003 Genetic and Evolutionary Computation Conference*, 2003.

[68] Hisao Ishibuchi and Youhei Shibata. Mating scheme for controlling the diversity-convergence balance for multi-objective optimization. In *The 2004 Genetic and Evolutionary Computation Conference*, volume 3102 of *Lecture Notes in Computer Science*. Springer, 2004.

[69] A. Jan, M. Yamamoto, and A. Ohuchi. Evolutionary algorithms for nurse scheduling problem. In *Proceedings of the 2000 Congress on Evolutionary Computation*, pages 196–203, 2000.

[70] Andrzej Jaszkiewicz. A metaheuristic approach to multiple objective nurse scheduling. *Foundations of Computing and Decision Sciences*, 22:169–184, 1997.

[71] Andrzej Jaszkiewicz. Genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research*, 137(1):50–71, 2002.

[72] Andrzej Jaszkiewicz. On the performance of multiple-objective genetic local search on the 0/1 knapsack problem - a comparative experiment. *IEEE Transactions on Evolutionary Computation*, 6(4):402–412, 2002.

[73] Huidong Jin and Man-Leung Wong. Adaptive diversity maintenance and convergence guarantee in multiobjective evolutionary algorithms. In *Proceedings of the 2003 Congress on Evolutionary Computation (CEC 2003)*, volume 4, pages 2498–2505. IEEE Press, 2003.

[74] H. Kawanaka, K. Yamamoto, T. Yoshikawa, T. Shinogi, and S. Tsuruoka. Genetic algorithm with the constraints for nurse scheduling problem. In *Proceedings of the 2001 Congress on Evolutionary Computation*, pages 1123–1130, 2001.

[75] Sami Khuri, Thomas Bäck, and Jörg Heitkötter. The zero/one multiple knapsack problem and genetic algorithms. In *Proceedings of the 1994 ACM Symposium on Applied Computing*, pages 188–193. ACM, 1994.

[76] Mifa Kim, Tomoyuki Hiroyasu, Mitsunori Miki, and Shinya Watanabe. Spea2+: Improving the performance of the strength pareto evolutionary algorithm 2. In *The 8th Parallel Problem Solving from Nature*, volume 3242 of *Lecture Notes in Computer Science*. Springer, 2004.

[77] Joshua Knowles and David Corne. Approximating the nondominated front using the pareto archived evolution strategy. *Evolutionary Computation*, 8:149–172, 2000.

[78] Joshua Knowles and David Corne. M-paea: A memetic algorithm for multi-objective optimization. In *Proceedings of the 2000 Congress on Evolutionary Computation*, volume 1, pages 325–332, 2000.

[79] Mario Koppen, Raul Vicente-Garcia, and Bertram Nickolay. Fuzzy-pareto-dominance and its application in evolutionary multi-objective optimization. In *Evolutionary Multi-Criterion Optimization Third International Conference (EMO 2005)*, volume 3410 of *Lecture Notes in Computer Science*, pages 399–412. Springer, 2005.

[80] Dario Landa-Silva and Edmund Burke. Using diversity to guide the search in multi-objective optimization. In Carlos Coello Coello and Gary Lamont, editors, *Applications of Multi-Objective Evolutionary Algorithms*, volume 1 of *Advances in Natural Computation*, pages 727–751. World Scientific, 2004.

[81] Dario Landa-Silva and Khoi Le. A simple evolutionary algorithm with self-adaptation for multi-objective nurse scheduling. In K. Srensen C. Cotta, M. Sevaux, editor, *Adaptive and Multilevel Metaheuristics*, volume 136 of *Studies in Computational Intelligence*, pages 133–155. Springer, 2008.

[82] Marco Laumanns, Lothar Thiele, Kalyamoy Deb, and Eckart Zitzler. Combining convergence and diversity in evolutionary multi-objective optimisation. *Evolution Computation*, 10:263–282, 2002.

[83] Marco Laumanns, Eckart Zitzler, and Lothar Thiele. On the effects of archiving, elitism, and density based selection in evolutionary multiobjective optimization. In *The 1st International Conference on Evolutionary Multi-Criterion Optimization*, volume 1993 of *Lecture Notes in Computer Science*. Springer, 2001.

[84] Khoi Le and Dario Landa-Silva. Obtaining better non-dominated sets using volume dominance. In *Proceedings of the 2007 Congress on Evolutionary Computation (CEC 2007)*, pages 3119–3126. IEEE Press, 2007.

[85] Khoi Le and Dario Landa-Silva. Adaptive and assortative mating scheme for evolutionary multi-objective algorithms. In *8th International Conference, Evolutionary Artificielle - EA 2007*, volume 4926 of *Lecture Notes in Computer Science*, pages 172–183. Springer, 2008.

[86] Khoi Le, Dario Landa-Silva, and Hui Li. An improved version of volume dominance for multi-objective optimisation. In *Proceedings of the 2009 International Conference on Evolutionary Multi-criterion Optimization*, volume 5467 of *Lecture Notes in Computer Science*, pages 231–245. Springer, 2009.

[87] H. Li, A. Lim, and B. Rodrigues. A hybrid ai approach for nurse rostering problem. In *Proceedings of the 2003 ACM Symposium on Applied Computing*, pages 730–735, 2003.

[88] Silvano Martello and Paolo Toth. *Knapsack Problems - Algorithms and Computer Implementations.* Wiley, 1990.

[89] A. Meisels, E. Gudes, and G. Solotorevsky. Employee timetabling, constraint networks and knowledge-based rules: A mixed approach. In *Procedings of the 1996 International Conference on the Practice and Theory of Automated Timetabling*, volume 1153 of *Lecture Notes in Computer Science*, pages 93–105, 1996.

[90] Zbigniew Michalewicz and Jaroslaw Arabas. Genetic algorithms for the 0/1 knapsack problem. In *Methodologies for Intelligent Systems*, volume 869 of *Lecture Notes in Computer Science*, pages 134–143. Springer, 1994.

[91] Kaisa Miettinen, Kalyanmoy Deb, Johannes Jahn, Wlodzimierz Ogryczak, Koji Shimoyama, and Rudolf Vetschera. Future challenges. In *Multiobjective Optimization: Interactive and Evolutionary Approaches*, volume 5252 of *Lecture Notes in Computer Science*, page 435461. Springer, 2008.

[92] H. E. Miller, W. Pierskalla, and G. Rath. Nurse scheduling using mathematical programming. *Operations Research*, 24:857–870, 1976.

[93] Julian Molina, Luis Santana, Alfredo Hernandez-Diaz, Carlos Coello Coello, and Rafael Caballero. g-dominance: Reference point based dominance for multiobjective metaheuristics. *European Journal of Operational Research*, 197:685692, 2009.

[94] Margarida Moz and Margarida Pato. A genetic algorithm approach to a nurse rerostering problem. *Computers & Operations Research*, 34:667–691, 2007.

[95] Christine Mumford. Comparing representations and recombination operators for the multi-objective 0/1 knapsack problem. In *Proceedings of the 2003 Congress on Evolutionary Computation*, volume 2, pages 854–861, 2003.

[96] Christine Mumford. Simple population replacement strategies for a steady-state multi-objective evolutionary algorithm. In *Genetic and Evolutionary Computation GECCO 2004*, volume 3102 of *Lecture Notes in Computer Science*, pages 1389–1400. Springer, 2004.

[97] N. Muslija, J. Gaertner, and W. Slany. Efficient generation of rotating workforce schedules. In *Proceedings of the 2000 International Conference on the Practive and Theory of Automated Timetabling*, 2000.

[98] I. M. Oliver, D. J. Smith, and J. R. C. Holland. A study of permutation crossover operators on the traveling salesman problem. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, pages 224–230, 1987.

[99] E. Özcan. Memetic algorithms for nurse rostering. In *The 2005 International Symposium on Computer and Information Sciences*, pages 482–492. Springer, 2005.

[100] I. Ozkarahan and J. E. Bailey. Goal programming model subsystem of a flexible nurse scheduling support system. *IEEE Transactions*, 16:306–316, 1988.

[101] Vilfredo Pareto. *Cours D'Economie Polotique.* F.Rouge, Lausanne, 1896.

[102] Jin Peng, Henry Mok, and Wai-Man Tse. Fuzzy dominance based on credibility distributions. In *Fuzzy Systems and Knowledge Discovery*, volume 3613 of *Lecture Notes in Computer Science*, pages 295–303. Springer, 2005.

[103] Sanja Petrovic, Gareth Beddoe, and Greet Vanden Berghe. Storing and adapting repair experiences in employee rostering. In *Practice and Theory*

of AutomatedTimetabling IV, volume 2740 of *Lecture Notes in Computer Science*, pages 148–165. Springer, 2003.

[104] Javier Puente, Alberto Gomez, Isabel Fernandez, and Paolo Priore. Medical doctor rostering problem in a hospital emergency department by means of genetic algorithms. *Computers & Industrial Engineering*, 56(4):1232–1242, 2009.

[105] Richard Rosenberg. *Simulation of Genetic Populations with Biochemical Properties.* PhD thesis, University of Michigan, Michigan, USA, 1967.

[106] Kumara Sastry, David Goldberg, and Graham Kendall. Genetic algorithms. In Edmund Burke and Graham Kendall, editors, *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, pages 97–125. Springer, 2005.

[107] David Schaffer. *Mutiple Objective Optimization with Vector Evaluated Genetic Algorithm.* PhD thesis, Vanderbilt University, Tennessee, USA, 1984.

[108] Jason Schott. Fault tolerant design using single and multicriteria genetic algorithm optimization. Master's thesis, The Department of Aeronautics and Astronautics, Massachesetts Institute of Technology, 1995.

[109] S. Scott and R. M. Simpson. Case-based incorporating scheduling constraint dimensions: Experiences in nurse rostering. In *Advances in Case-Based Reasoning*, volume 1488, pages 392–401. Springer, 1998.

[110] N Srinivas and Kalyamoy Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2:221–248, 1994.

[111] T. Starkweather, S. McDaniel, D. Whitley, K. Mathias, and Whitley C. A comparison of genetic sequencing oper. In *Proceedings of the 1991 International Conference on Genetic Algorithms*, 1991.

[112] G. Syswerda. Handbook of genetic algorithms schedule optimization using genetic algorithms. In L. David, editor, *Handbook of Genetic Algorithms*, pages 332–349. Van Nostrand Reinhold, New York, 1991.

[113] Lothar Thiele, Kaisa Miettinen, Pekka Korhonen, and Julian Molina. A preference-based evolutionary algorithm for multi-objective optimization. *Evolution Computation*, 17(3):411–436, 2009.

[114] V. M. Trivedi and M. Warner. A branch and bound algorithm for optimum allocation of float nurses. *Management Science*, 22:972–981, 1976.

[115] Christine Valenzuela. A simple evolutionary algorithm for multi-objective optimization (seamo). In *Proceedings of the 2002 Congress on Evolutionary Computation - CEC 2002*, pages 717–722, 2002.

[116] M. Warner. Nurse staffing, scheduling, and reallocation in the hospital. *Hopistal and Health Services Administration*, 2:77–90, 1976.

[117] M. Warner and J. Prawda. A mathermatical programming model for scheduling nursing personnel in a hospital. *Management Science*, 19:411–422, 1972.

[118] Upali Wickramasinghe, Robert Carrese, and Xiaodong Li. Designing airfoils using a reference point based evolutionary many-objective particle swarm optimization algorithm. In *Proceedings of the 2010 Congress on Evolutionary Computation*, pages 1857 – 1864. IEEE, 2010.

[119] Jin Wu and Shapour Azarm. Metrics for quality assessment of a multi-objective design optimization solution set. *Journal of Mechanical Design*, 123:18–25, 2001.

[120] P. L. Yu. Cone convexity, cone extreme points, and nondominated solutions in decision problems with multiobjectives. *Journal of Optimization Theory and Applications*, 14(3):319–377, 1974.

[121] Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[122] Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications.* PhD thesis, Computer Engineering and Networks Laboratory, Swiss Federal Institute of Technology Zurich, 1999.

[123] Eckart Zitzler and Simon Künzli. Indicator-based selection in multiobjective search. In *Parallel Problem Solving from Nature (PPSN VIII)*, pages 832–842. Springer-Verlag, 2004.

[124] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization. In *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. International Center for Numerical Methods in Engineering (CIMNE), 2002.

[125] Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

Appendix - List of Publications

Khoi Le and Dario Landa-Silva. Obtaining better non-dominated sets using volume dominance. In *Proceedings of the 2007 Congress on Evolutionary Computation (CEC 2007)*, pages 3119-3126. IEEE Press, 2007.

Khoi Le and Dario Landa-Silva. Adaptive and assortative mating scheme for evolutionary multi-objective algorithms. In *8th International Conference, Evolutionary Artificielle - EA 2007*, volume 4926 of *Lecture Notes in Computer Science*, pages 172-183. Springer, 2008.

Khoi Le, Dario Landa-Silva, and Hui Li. An improved version of volume dominance for multi-objective optimisation. In *Proceedings of the 2009 International Conference on Evolutionary Multi-criterion Optimization*, volume 5467 of *Lecture Notes in Computer Science*, pages 231–245. Springer, 2009.

Khoi Le and Dario Landa-Silva. Hyper Volume Evolutionary Algorithm. *Preparing for submission.* 2010.