EVOLUTIONARY MULTI-OBJECTIVE OPTIMIZATION IN SCHEDULING PROBLEMS

CHEONG CHUN YEW

B.Eng (Hons., 1st Class), NUS

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2009

Abstract

The primary aim of this thesis is to present an investigation on the application of multi-objective evolutionary algorithms (MOEAs) to solve a few real-world scheduling problems with vastly different characteristics. Real-world scheduling problems are generally complex, large scale, constrained, and multi-objective in nature that classical operational research techniques are inadequate at solving them effectively. Optimal solutions to these problems in today's productivity-oriented world would have significant economic and social consequences. In this thesis, a generic MOEA framework is devised and problem-specific operators are then designed to adapt the MOEA to solve the different scheduling problems. The research documented in this thesis represents one of the pioneering works on multi-objective optimization of each of the scheduling problems investigated.

One of the scheduling problems considered in this thesis is a two-objective exam timetabling problem (ETTP), which involves the scheduling of exams for a set of university courses into a timetable such that there are as few occurrences of students having to take exams in consecutive periods as possible but at the same time minimizing the timetable length and satisfying hard constraints such as limited seating capacity and no overlapping exams. While existing approaches require prior

knowledge of the timetable length in order to be effective, the MOEA proposed in this thesis provides a more general solver to the ETTP by including the timetable length as an optimization objective.

A berth allocation problem (BAP), which requires the determination of exact berthing times and positions of incoming ships in a container port, is also studied in this thesis. The BAP considers three objectives of minimizing makespan, waiting time, and degree of deviation from a predetermined priority schedule, which represent the interests of both port and ship operators. The experimental results reveal several interesting relationships between the objectives, justifying the multi-objective approach to the problem, which has never been explored for this problem.

This thesis also considers a three-objective vehicle routing problem with stochastic demand (VRPSD), which involves the routing of a set of identical vehicles with limited capacity from a central depot to a set of geographically dispersed customers to satisfy their demands. Unlike the ETTP and the BAP, where all aspects of the problem are known at the point of solving the problem, the VRPSD is a stochastic optimization problem and some problem parameters are uncertain during the solution-searching process. In the VRPSD, the actual demand of each customer is unknown during the routing process but is revealed only when the vehicle reaches the customer. The experimental results show that the solutions obtained by the MOEA are robust to the stochastic nature of the problem.

Acknowledgements

First and foremost, I would like to thank my Ph.D. supervisor, Associate Professor Tan Kay Chen for introducing me to the wonderful field of computational intelligence and giving me the opportunity to pursue research. His indispensable guidance and advices, both academically and personally, have kept my work on course during the past four years.

I am also grateful to my fellow lab buddies at the Control and Simulation Laboratory, who have one way or another made my Ph.D. life very enjoyable: Chi Keong for being the Grand Jedi Master of our research group, Dasheng for maintaining the group server, Eujin for looking after the lab after we have all gone home, Brian for providing lunch ferry service in his Man U Mobile, Hanyang for accompanying me from the first day till the last, Chiam for bringing me into the world of "software beta testing", Chin Hiong for providing a venue for our group's research on "dry swimming", and Chen Jia and Vui Ann for being the replacements. I would also wish to express my gratitude to the lab officers, including Hengwei, Sara, and Chee Siong, for the assistance provided during my time in the lab.

Last but not least, I would like to thank my family for all their love and care. This thesis would not have been possible without their consistent support.

Publications

Journals

K. C. Tan, C. Y. Cheong, and C. K. Goh, "Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation", *European Journal of Operational Research*, vol. 177, no. 2, pp. 813 – 839, 2007.

C. Y. Cheong, K. C. Tan, and B. Veeravalli, "A multi-objective evolutionary algorithm for examination timetabling", *Journal of Scheduling*, vol. 12, no. 2, pp. 121 – 146, 2009.

C. Y. Cheong, K. C. Tan, D. K. Liu, and C. J. Lin, "Multi-objective and prioritized berth allocation in container ports", *Annals of Operations Research*, in press.

C. K. Goh, K. C. Tan, C. Y. Cheong, and Y. S. Ong, "An investigation on noise-induced features in robust evolutionary multi-objective optimization", *Expert Systems with Applications*, in press.

K. C. Tan, C. Y. Cheong, and Y. Peng, "A genetic algorithm approach for real-time identification and control of a helicopter system", *International Journal of Innovative Computing*, *Information and Control*, submitted.

Conference Papers

C. Y. Cheong, K. C. Tan, D. K. Liu, and J. X. Xu, "A multiobjective evolutionary algorithm for solving vehicle routing problem with stochastic demand", in *Proceedings of the IEEE Congress on Evolutionary Computation 2006*, *CEC 2006*, Vancouver, Canada, pp. 5519 – 5526, 2006.

C. Y. Cheong, K. C. Tan, and B. Veeravalli, "Solving the exam timetabling problem via a multi-objective evolutionary algorithm – a more general approach", in *Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Scheduling*, *CI-Sched 2007*, Honolulu, HI, USA, pp. 165 – 172, 2007.

C. Y. Cheong, S. C. Chiam, and C. K. Goh, "Eliminating positional dependency in binary representation via redundancy", in *Proceedings of the 2007 IEEE Symposium on Foundations of Computational Intelligence*, *FOCI 2007*, Honolulu, HI, USA, pp. 251 – 258, 2007.

C. Y. Cheong, C. J. Lin, K. C. Tan, and D. K. Liu, "A multi-objective evolutionary algorithm for berth allocation in a container port", in *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*, *CEC 2007*, pp. 927-934, 2007.

C. K. Goh, K. C. Tan, C. Y. Cheong, and Y. S. Ong, "Noise-induced features in robust multi-objective optimization problems", in *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*, *CEC 2007*, pp. 568-575, 2007.

C. Y. Cheong, K. C. Tan, and D. K. Liu, "Solving the berth allocation problem with service priority via multi-objective optimization", in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Scheduling*, *CI-Sched 2009*, Nashville, TN, USA, pp. 95 – 102, 2009.

Book Chapters

C. Y. Cheong and K. C. Tan, "A multi-objective multi-colony ant algorithm for solving the berth allocation problem", *Advances of Computational Intelligence in Industrial Systems*, Y. Liu, A. Sun, H. T. Loh, W. F. Lu, and E.-P. Lim (Eds.), Springer-Verlag, pp. 333 – 350, 2008.

C. Y. Cheong and K. C. Tan, "Hybridizing problem-specific operators with meta-heuristics for solving the multi-objective vehicle routing problem with stochastic demand", *Bio-Inspired Approaches for the Vehicle Routing Problem*, F. B. Pereira and J. Tavares (Eds.), Springer-Verlag, pp. 101 – 129, 2009.

Contents

List of Figures

List of Tables

List of Abbreviations

BAP	Berth allocation problem
CCP	Chance constrained program
CTTP	Course timetabling problem
DET	Deterministic approach
EA	Evolutionary algorithm
ε-MOEA	ε-multi-objective evolutionary algorithm
ETTP	Exam timetabling problem
FastPGA	Fast Pareto genetic algorithm
FCFS	First-come-first-serve
GENMOP	General multi-objective parallel genetic algorithm
GRASP	Greedy randomized adaptive search procedures
IMOEA	Incrementing multi-objective evolutionary algorithm
LCFS	Last-come-first-serve
MGA	Micro-genetic algorithm
MOEA	Multi-objective evolutionary algorithm
MOGA	Multi-objective genetic algorithm
mohBOA	Multi-objective hierarchical Bayesian optimization algorithm
MOMGA	Multi-objective messy genetic algorithm
NPGA	Niched Pareto genetic algorithm

NSGA	Non-dominated sorting genetic algorithm
OmniOpt	Omni-optimizer
PAES	Pareto archived evolution strategy
PCGA	Pareto converging genetic algorithm
PESA	Pareto envelope-based selection algorithm
RSM	Route simulation method
SPEA	Strength Pareto evolutionary algorithm
SPR	Stochastic program with recourse
SPS	Shortest path search
SVRP	Stochastic vehicle routing problem
Type-C	Test problem with clustered customers
Type-CS	Stochastic version of Type-C
Type-R	Test problem with remote customers
Type-RC	Test problem with remote and clustered customers
Type-RCS	Stochastic version of Type-RC
Type-RS	Stochastic version of Type-R
VEGA	Vector evaluated genetic algorithm
VRP	Vehicle routing problem
VRPSD	Vehicle routing problem with stochastic demand
VRPTW	Vehicle routing problem with time windows
WDS	Which directional search

Chapter 1

Introduction

1.1 Background

In everyday life, we are often confronted with a variety of optimization problems which require decisions to be made so as to obtain the best attainable results out of limited available resources. Some examples include deciding what type of transport to take, what household chore to be done next, and what groceries to buy. For these routine tasks, the decision to be made for, say, the cheapest form of transportation to get to our destination can be very obvious. Consider now the situation where we are running late for a meeting due to some unforeseen circumstances. Since the need for expedition is conflicting to the first consideration of minimizing cost, the selection of the right form of transportation is no longer as straightforward as before and the final solution will represent a compromise between the two objectives. This type of problems, which involves the simultaneous consideration of multiple conflicting objectives, is commonly termed as multi-objective problems.

In a single-objective optimization problem, the notion of optimality is straightforward. The best solution is the one that realizes the minimum or the maximum of the objective function. However, in a multi-objective optimization problem, the notion of optimality is not that obvious. Since no one solution can be termed as optimal in the face of multiple conflicting objectives, the goal of multi-objective optimization lies in finding the set of tradeoff solutions that is better than the other solutions in the entire search space when considering all the objectives. To be specific, within this set of tradeoff solutions, known in the literature as the Pareto-optimal set, no one solution is better than any other solution in terms of the multiple objectives. For any solution in the search space not in the Pareto-optimal set, there is at least one solution in the Pareto-optimal set that is better than the former in terms of all the objectives. Based on the Pareto-optimal set, the decision maker can then make an informed decision on which of the tradeoff solutions to pick for actual implementation. This sums up the whole solution process for multi-objective optimization.

1.2 Motivation

Multi-objective optimization problems can be found in various fields, including engineering, bioinformatics, logistics, economics, finance, or wherever optimal decisions need to be made in the presence of tradeoffs between two or more conflicting objectives. This research investigates multi-objective optimization in scheduling problems.

1.2.1 Multi-Objective Optimization in Scheduling Problems

Scheduling can be regarded as a decision making process which involves the allocation of limited resources to tasks over time. One of the more popular definitions of scheduling was given by Wren (1996), who stated that "Scheduling may be seen as the arrangement of objects into a pattern in time or space in such a way that some goals are achieved, or nearly achieved, and that constraints on the way the objects may be arranged are satisfied, or nearly satisfied". From the definition of Wren (1996), it is clear that scheduling problems are typically characterized by a number of goals (or objectives) and constraints. It can also be seen from the definition that it may not always be possible for all the constraints in scheduling problems to be completely satisfied. This leads to the classification of scheduling problem constraints into hard and soft constraints based on their criticality. Hard constraints are those that must be satisfied at all cost in order for the schedule to be feasible. Failure to completely satisfy this class of constraints would render the schedule useless. On the other hand, the satisfaction of soft constraints is considered desirable but it is not absolutely essential for the complete satisfaction of this class of constraints. In fact, the satisfaction of soft constraints is typically modeled as the objectives of scheduling problems such that the number of soft constraint violations is required to be minimized. As such, given that the objectives of scheduling problems include their original objectives as well as the minimization of soft constraint violations, they are naturally multi-objective optimization problems.

1.2.2 Multi-Objective Evolutionary Algorithms

In this research, evolutionary algorithms (EAs) are applied for multi-objective optimization in scheduling problems. EAs are a class of stochastic optimization techniques introduced in the 1960s by Fogel *et al*. (1966) and in the 1970s by Rechenberg (1973) and Holland (1975). EAs work by simulating biological evolution. They operate on a population of candidate solutions that increasingly adapts to the problem domain through an iterative process of biologically inspired operators, including selection, crossover, and mutation. They have the capability to produce near-optimal, if not exact-optimal, solutions for multi-dimensional problems and thus have been successfully applied to a wide variety of problems (Ross and Corne, 1994). An EA that is employed in the multi-objective optimization context is known in the literature as a multi-objective evolutionary algorithm (MOEA).

1.2.3 Why are Evolutionary Algorithms Suitable for Multi-Objective Problems

The classical approach to a multi-objective optimization problem involves forming an aggregate objective function based on the weighted sum of the objectives, where the weight associated with an objective is proportional to the preference assigned to that particular objective. This method effectively converts the multi-objective problem into a single-objective one. The optimization based on this aggregate objective function may then lead the search to one of the tradeoff solutions in the Pareto-optimal set. The solution obtained using this approach is highly dependent on the weight vector used in forming the aggregate objective function. Changing the weight

vector may (or may not) yield another solution in the Pareto-optimal set. Another problem with this approach is that the process of finding an appropriate weight vector is highly subjective. It requires an analysis of non-technical, qualitative, and experience-driven information to find a quantitative weight vector representing the preferences of the decision maker (Deb, 2001). Moreover, the process has to be carried out without any knowledge of the likely set of tradeoff solutions or how the multiple objectives are related to one another.

Although the classical multi-objective optimization approach described above has a number of deficiencies, it is not difficult to understand that its development was motivated by the fact that classical optimization techniques are designed to find a single solution in each simulation run. Such techniques use a point-to-point approach, which involves searching iteratively from an incumbent solution to its neighborhood, and are capable of generating only one solution per simulation run. As such, there was a need to convert the task of finding multiple tradeoff solutions of a multi-objective problem to one of finding a single solution of a transformed single-objective problem. However, with the advent of EAs in recent years, the landscape of the field of optimization has changed drastically. The most prominent difference between EAs and classical optimization techniques is that EAs operate on a population of candidate solutions and their end product is also a population of solutions. If an EA is applied to a single-objective problem, one can expect the population of solutions to converge to the optimal solution. On the other hand, if the problem has more than one optimal solution, the EA can capture the multiple solutions in its final population. This ability of EAs to find multiple optimal solutions

in a single simulation run makes them natural solvers of multi-objective optimization problems.

1.2.4 Why are Evolutionary Algorithms Suitable for Scheduling Problems

Scheduling problems are well-known to be NP-complete (Garey and Johnson, 1979; Karp, 1972). This means that there is no known algorithm that is capable of finding optimal solutions to scheduling problems in polynomial time. Even though there are exact algorithms that guarantee finding optimal solutions to some simplified forms of scheduling problems, these approaches generally take too long to generate meaningful solutions when the problem size gets larger or when additional constraints are added.

Solving scheduling problems is not a new research topic. Many solution methods have been proposed and implemented. Early approaches solved simplified versions of the problem exactly. However, it soon became apparent that real-world scheduling problems are so large and complex that it is simply impossible to consider every single solution in the search space to find exact solutions. As a result, focus was shifted to designing heuristic methods to find good, near-optimal, solutions or to simply find feasible solutions for the really difficult problems. Most research now involves designing better heuristics for specific instances of scheduling problems. However, such heuristic methods are typically limited to a specific set of constraints or problem formulation. The complex and combinatorial nature of scheduling problems then led many researchers to experiment with EAs as a solution method.

EAs are well-known for their ability to solve non-linear and combinatorial problems. They are also often noted for searching large, multi-modal spaces effectively since they operate on a population of solutions, which allows them to sample multiple candidate solutions simultaneously. Unlike exact algorithms, EAs do not promise optimal solutions but they focus their search on more promising areas in the search space, allowing them to find near-optimal solutions within acceptable time. EAs also do not require any gradient or problem-specific information, making them a more general solver of scheduling problems compared to heuristic methods.

1.3 Organization of this Thesis

The suitability of EAs to solve multi-objective scheduling problems presented in this chapter provided the main motivation for the research documented in this thesis. The primary aim of this thesis is to present an investigation on the application of MOEAs to solve a few scheduling problems with vastly different characteristics. A generic MOEA framework will first be devised. Problem-specific operators are then designed to adapt the MOEA to solve the different scheduling problems considered in this thesis.

The organization of the remaining portion of this thesis is as follows. Chapter 2 provides a brief review of multi-objective optimization and MOEAs. Basic concepts of multi-objective optimization, including Pareto dominance and Pareto optimality, are introduced. Some MOEA design issues are also highlighted. The chapter also describes several state-of-the-art MOEAs and their features for handling multi-

objective optimization. Chapter 3 presents the framework of the generic MOEA that will be applied to solve three very different scheduling problems in this thesis. The program flow and several problem-independent components of the MOEA are described in detail.

Chapter 4 considers the application of the MOEA on a two-objective exam timetabling problem (ETTP). The ETTP involves the scheduling of exams for a set of university courses into a timetable such that there are as few occurrences of students having to take exams in consecutive periods as possible but at the same time minimizing the timetable length and satisfying hard constraints such as limited seating capacity and no overlapping exams.

Chapter 5 studies a berth allocation problem (BAP) which requires the determination of exact berthing times and positions of incoming ships in a container port. Unlike the two-objective ETTP, the BAP considers three objectives of minimizing makespan, waiting time, and degree of deviation from a predetermined priority schedule. These objectives represent the interests of both port and ship operators.

A multi-objective vehicle routing problem with stochastic demand (VRPSD) is considered in Chapter 6. The VRPSD involves the routing of a set of identical vehicles with limited capacity from a central depot to a set of geographically dispersed customers to satisfy their demands. Unlike the ETTP and the BAP, where all aspects of the problem are known at the point of solving the problem, the VRPSD is a stochastic optimization problem and some problem parameters are uncertain during the solution-searching process. In the VRPSD, the actual demand of each

customer is unknown during the routing process but is revealed only when the vehicle reaches the customer.

Finally, the contributions of this thesis and some directions for future work are discussed in Chapter 7.

Chapter 2

A Review of Multi-Objective Evolutionary Algorithms

2.1 Basic Concepts of Multi-Objective Optimization

In real-world problems, the quality of a solution can rarely be measured by a single criterion. In fact, several criteria are usually used to gauge the quality of a solution and these criteria have different nature and importance and are usually conflicting with one another, i.e. an improvement in one of the criteria can only be achieved at the expense of worsening another. In many cases, the criteria are also incommensurable, i.e. there is no common standard of comparison for the criteria. This gives rise to the need for effective multi-objective optimization techniques that are able to generate solutions that respect the various criteria of a problem.

There are generally three approaches to multi-objective optimization in the literature (Goicoechea *et al*., 1982; Steuer, 1986).

1) Combining the objectives: As mentioned in the introduction, this is one of the classical approaches to multi-objective optimization. It involves forming an aggregate objective function based on the weighted sum of the objectives and converts the multi-objective problem into a single-objective one. Although the approach is simple and allows existing single-objective algorithms to be directly applied to solve the problem, the optimization outcome is highly susceptible to the choice of weights used in aggregating the various objectives.

2) Optimizing one objective at a time: This approach involves optimizing with respect to one objective at a time while imposing constraints on the other objectives. The problem with this approach is that the optimization outcome is highly dependent on the order in which the objectives are considered for optimization.

3) Optimizing all objectives simultaneously: This approach, also known as Pareto optimization, uses the concept of Pareto dominance, which was formulated by the French economist Vilfredo Pareto (1848 – 1923), to compare the optimality of solutions.

The first two approaches require preference information from the decision maker before they perform the search process and are known as *a priori* approaches. On the other hand, Pareto optimization, which is the main approach studied in this thesis, is an *a posteriori* approach that does not depend on the decision maker's preferences. It aims to find the set of Pareto-optimal solutions from which the decision maker can choose the most preferable one. The strategies that a decision maker uses to pick a solution from the Pareto-optimal set is studied in another field known as multi-attribute decision making (Vincke, 1992), which is out of the scope of this thesis.

2.1.1 Pareto Dominance and Optimality

The concepts of Pareto dominance and Pareto optimality are fundamental in the Pareto optimization approach to multi-objective problems, with Pareto dominance forming the basis for solution quality comparison.

Consider two distinct vectors $\mathbf{U} = (u_1, u_2, u_3, \ldots, u_k)$ and $\mathbf{V} = (v_1, v_2, v_3, \ldots, v_k)$ representing the objective values of two solutions for a k-objective minimization problem. There are three possible relationships between the two solutions, which are defined by Pareto dominance (Dasgupta *et al*., 1999; Van Veldhuizen and Lamont, 2000; Zitzler, 1999):

- **Strong dominance**: **U** strongly dominates **V** (denoted by $\mathbf{U} \prec \mathbf{V}$) if $u_i < v_i$, for $i = 1, 2, 3, \ldots, k$.
- **Weak dominance**: **U** weakly dominates **V** (denoted by $\mathbf{U} \preceq \mathbf{V}$) if $u_i \leq v_i$, for $i = 1, 2, 3, \ldots, k$ and $u_i < v_i$, for at least one i.
- **Incomparable**: **U** and **V** are incomparable (denoted by $\mathbf{U} \sim \mathbf{V}$) if neither **U** (strongly or weakly) dominates **V** nor **V** (strongly or weakly) dominates **U**.

Fig. 2.1 provides an illustration of the three Pareto dominance relationships highlighted above for a two-objective example. With solution A as the point of reference, the regions highlighted in different shades of grey in the figure represent the three different dominance relations. Solutions located in the dark grey region are strongly dominated by solution A because A is better in both objectives. For the same reason, solutions located in the white region strongly dominate solution A. Although A has a smaller objective value as compared to the solutions located at the boundaries

between the dark and light grey regions, it only weakly dominates these solutions by virtue of the fact that they share a similar objective value along either one dimension. Solutions located in the light grey regions are incomparable to solution A because it is not possible to establish any superiority of one solution over the other since the solutions in the left light grey region are better only in the second objective while the solutions in the right light grey region are better only in the first objective.

In this thesis, weak dominance is used to distinguish the quality of two solutions, i.e. as long as a solution weakly dominates another solution, it is considered to be the better solution (out of the two). For convenience, weak dominance will be referred to as dominance in the rest of this thesis.

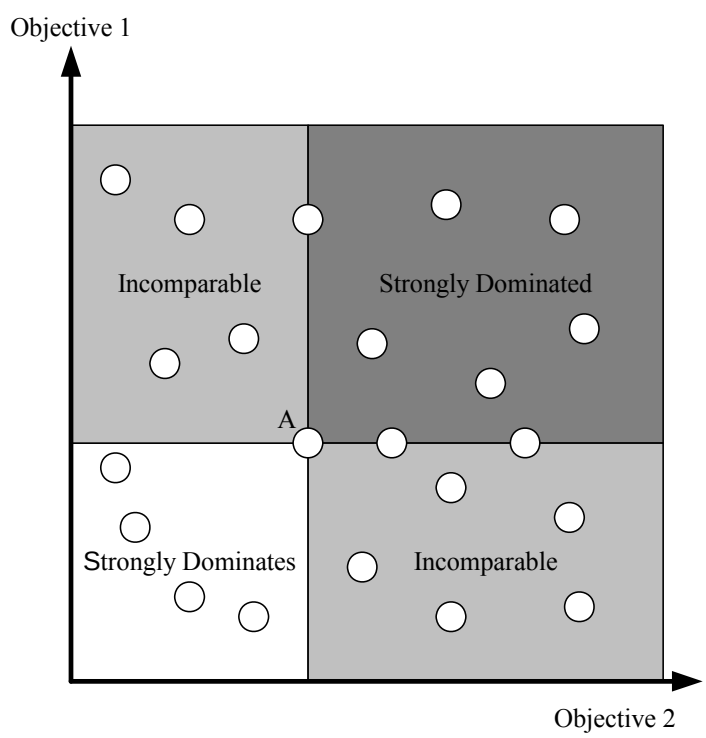


Fig. 2.1 Illustration of Pareto dominance relationship

With the definition of Pareto dominance, the set of solutions desirable for multi-objective optimization can now be more formally defined. A solution x is said to be non-dominated with respect to a set of solutions S if there is no other solution in S that dominates x, although it is likely that there are solutions in S that are incomparable to x. Based on this concept of non-dominance, it is clear that the aim of multi-objective optimization is to find the set of all non-dominated solutions in the entire search space. As mentioned in the introduction, this set of solutions is known as the Pareto-optimal set. All the solutions in the Pareto-optimal set are incomparable with one another and for any solution in the search space not in the Pareto-optimal set, there is at least one solution in the Pareto-optimal set that dominates the former. The solutions in the Pareto-optimal set compose a boundary between the space which contains the dominated solutions and the infeasible region where no solution exists. This boundary is known as the tradeoff surface or the Pareto-optimal front. It can be depicted as a hyperplane in the k-dimensional space, where k is the number of objectives. For a two-objective example, shown in Fig. 2.2, the Pareto-optimal front is presented as a curve. It can also be seen from Fig. 2.2 that each objective component of any solution in the Pareto-optimal set can only be improved by degrading at least one of its other objective components (Srinivas and Deb, 1994).

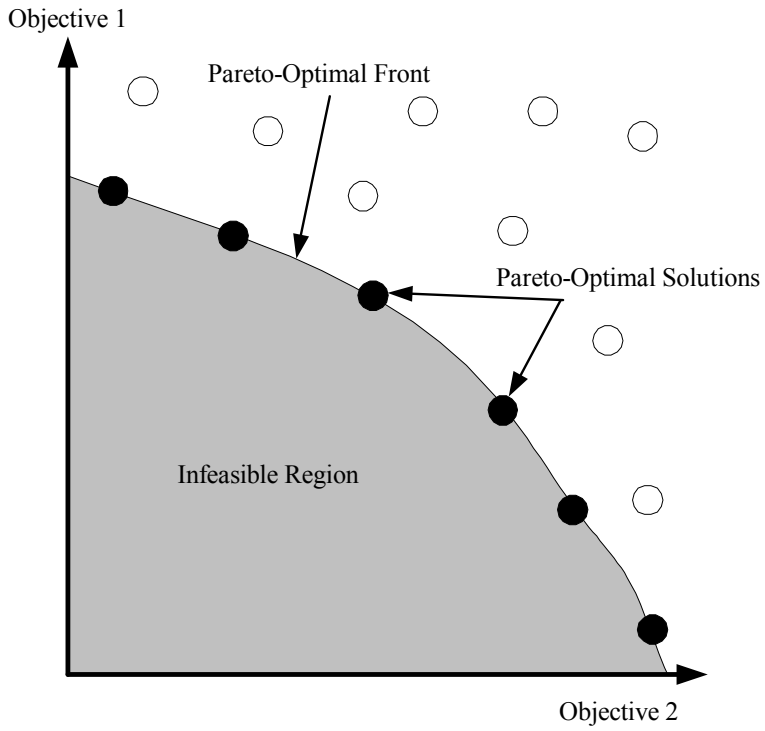


Fig. 2.2 Illustration of Pareto-optimal front

2.1.2 Quality of an Obtained Pareto Front

The Pareto-optimal set and Pareto-optimal front introduced in the previous section represent an ideal solution that a multi-objective optimization algorithm should aspire to achieve. However, due to the complexity of real-world problems, one can only hope to obtain a Pareto set of solutions (also referred to as Pareto solutions) that can approximate the Pareto-optimal set as much as possible, i.e. the corresponding Pareto front obtained should be as close as possible to the Pareto-optimal front. Furthermore, in many real-world problems, there is no knowledge of the localization of the Pareto-optimal set or the shape of the Pareto-optimal front. As such, there is a need to define

some criteria to determine how good an obtained Pareto set of solutions is. These criteria are listed below (Deb, 2001; Zitzler, 1999):

- The closeness between the obtained Pareto front and the Pareto-optimal front (assuming the Pareto-optimal front is known).
- A good distribution of solutions along the obtained Pareto front.
- A wide spread of solutions along the obtained Pareto front.
- Maximize the number of Pareto solutions obtained.

While the first criteria of getting solutions that are as close as possible to the optimal solutions is the primary consideration of all optimization problems, the remaining criteria are unique to multi-objective optimization and they sought to obtain a diverse set of solutions. The rationale of finding a diverse and uniformly distributed set of solutions is to provide the decision maker with sufficient information about the tradeoffs between the different solutions before the final decision is made. It should also be noted that some of the criteria listed above are conflicting in nature, which further explains why multi-objective optimization is much more challenging than single-objective optimization.

2.2 Multi-Objective Evolutionary Algorithms

The EA is one of the first meta-heuristics to be adapted for multi-objective optimization (Van Veldhuizen and Lamont, 2000) due to its population-based nature, which makes EAs well-suited for finding multiple tradeoff solutions in a multi-objective problem. In this section, the functions of the different components of an EA

are discussed and a brief review of some of the more representative MOEAs in the literature is provided.

2.2.1 Evolutionary Algorithms

The EA is a general purpose optimization tool inspired by Darwin's theory of evolution (Goldberg, 1989; Michalewicz, 1999). The basic idea in EAs is to generate a population of individuals (representing a population of candidate solutions) and evolve this population, by means of selection, recombination, and mutation, over a number of generations. Fig. 2.3 shows the pseudo-code of a typical EA. Evolution is driven by a selection mechanism, which is based on the principle of survival-of-the-fittest (Dawkins, 1976) and marks fitter individuals (higher quality candidate solutions) as parents. Recombination is implemented by a crossover operator, which combines two or more parents to form one or more offspring (new candidate solution). On the other hand, self-adaptation is implemented by a mutation operator, which makes small random perturbations to offspring. The selection mechanism serves to ensure that better candidate solutions participate to generate the next generation of (hopefully even better) solutions. The purpose of crossover is to propagate good solution components from parent solutions to their offspring, while the purpose of mutation is to add diversity to the population of candidate solutions. The three operators of selection, crossover, and mutation are fundamental to any EA and each operation of the trio represents the passing of a generation. After a series of improvement in every generation, at the termination of the EA when the stopping

criterion is satisfied, it is expected that the population of candidate solutions converge to a set of high quality solutions.

```
Generate initial population;
REPEAT
    Evaluate each individual in the population;
    Select individuals to act as parents;
    Apply Crossover to parents to create offspring;
    Apply Mutation to offspring;
    Select parents and offspring to form the new population;
UNTIL stopping criterion is satisfied;
```

Fig. 2.3 Pseudo-code of a typical EA

Designing an effective EA involves the careful selection of the following components.

1) Solution representation: The representation of solutions as individuals (or chromosomes due to the evolutionary operators of crossover and mutation having roots in the field of biology) is one of the most important issues in designing an EA. The choice of representation fundamentally influences the design of the other components in the EA. A good representation helps to ensure that the entire search space can be explored as much as possible. There are generally three types of representation. Direct representation, such as permutation-based representation (Carretero *et al*., 2007; Middendorf *et al*., 2002; Prins, 2000) and table/matrix representation (Hu and Di Paolo, 2009; Kacem *et al*., 2002; Miwa *et al*., 2002), encodes solutions in a straightforward way. Indirect representation (Aickelin and Dowsland, 2004; Cowling *et al*., 2002; Hindi *et al*., 2002) requires additional steps to generate the final solutions from the chromosomes. Rule-based representation

(Jahangirian and Conroy, 2000; Su and Shiue, 2003; Tay and Ho, 2008) involves using the EA to evolve the rules for constructing the actual solutions.

2) Selection mechanism: Unlike solution representation, choosing an appropriate selection mechanism is less problem-dependent since its main purpose is to distinguish the better solutions from a population of solutions. As such, most EAs use one of the several prescribed selection mechanisms available in the literature (Coley, 1999). One of these methods is the fitness-proportionate selection scheme, where the probability of an individual being selected to be a parent is proportional to its fitness. Another common selection mechanism is the tournament selection scheme, where the population is divided into groups and the individuals within each group compete to be selected as parents. One of the main design considerations of a selection mechanism is its selection intensity (Vajda *et al*., 2008). While it is usually acceptable for a selection scheme to always pick the best solutions as parents, one has to be careful of it driving the EA towards premature convergence. Furthermore, some inferior individuals may have useful solution components which may lead the search towards the optimal solutions. As such, it is recommended that a selection scheme offers a small non-zero chance that inferior solutions get selected as parents as well.

3) Crossover: The idea of crossover operation is similar to mating behavior in nature. In most EAs, two parents are selected from the population and new individuals are created by taking information from both of the parents. This interaction can be perceived as an information exchange session among different individuals in a society. The crossover operator has evolved from the traditional single-point crossover into a variety of interesting procedures today. Some of the

more popular crossover operators that have been applied in scheduling problems include order crossover (Goldberg, 1989; Wang and Zheng, 2003), cycle crossover (Hussain *et al*., 2002; Michalewicz, 1999; Moraglio *et al*., 2006), partial mapping crossover (Goh *et al*., 2003; Sahu and Tapadar, 2007; Wang and Zheng, 2003), and edge crossover (Hussain *et al*., 2002; Ponnambalam *et al*., 2002; Sokolov *et al*., 2005). Choosing a suitable crossover operator is one of the key factors that will determine the quality of optimization results (Deb and Beyer, 2001; Deb *et al*., 2002a).

4) Mutation: In contrast to crossover, mutation is a unary operator that involves only a single individual. The initial aspiration of using mutation is to prevent the EA from converging onto a local optimum in the search space. The rate at which mutation is applied to offspring is usually set to a small number as high mutation activity would destroy the convergence behavior of the optimization process. As such, the mutation rate is an important design parameter that has to be chosen carefully. Some popular mutation operators that have been applied in scheduling problems are swap mutation (Shaw and Fleming, 2000; Shrivastava and Dhingra, 2002; Zhang *et al*., 2006), swift mutation (Burdett and Kozan, 2000; Puljic and Manger, 2005), insertion mutation (Basseur *et al*., 2002; Ishibuchi *et al*., 2003; Oğuz and Ercan, 2005), and order mutation (Hart *et al*., 1999; Varela *et al*., 2003).

5) Constraint handling: In constrained problems, such as scheduling problems, it is very likely that the application of the evolutionary operators of crossover and mutation would generate infeasible solutions. Although a careful selection of the solution representation or a creative design of the evolutionary operators may allow

the EA to operate within feasible regions of the search space, this is not possible in most problems. The choice then is either to allow constraint violations but penalize them in the objective function or to reject the infeasible solution and apply the evolutionary operators repeatedly until a feasible solution is achieved or to design repair heuristics to search for a feasible alternative to the infeasible solution. Each of these approaches has its pitfalls. The first approach does not force the search to feasible regions of the search space and it is likely that the algorithm would waste computation effort searching within the infeasible regions, while the other two approaches may excessively increase the computation time of the algorithm due to the need to find a feasible solution each time an infeasible solution is encountered. An effective design of the constraint handling features in an EA is pertinent to the success of the algorithm.

6) Elitism: The way in which the offspring and parents combine to form the new population for the next generation is another design consideration that has a direct effect on the optimization performance of an EA. A non-elitist strategy replaces all individuals in the current population while an elitist one always keeps the best solutions found to date in the population. The former approach may result in a slow convergence while the latter may cause the search to be trapped in a local optimum.

From the various EA design considerations discussed above, it can be seen that there are many challenges involved in designing an effective EA. Some of these challenges involve solving multi-objective problems themselves. After deciding on the design of the various components of an EA, there is also a need to fine-tune the

various parameters, such as crossover rate, mutation rate, and population size, associated with the EA.

2.2.2 State-of-the-Art Multi-Objective Evolutionary Algorithms

In this section, six popular MOEAs, with various features for handling multi-objective optimization problems and maintaining population distribution on the tradeoff surface, are briefly described and discussed in chronological order.

1) Vector evaluated genetic algorithm (VEGA): The VEGA, proposed by Schaffer in 1985 (Schaffer, 1985), is widely recognized as the first MOEA to be developed. VEGA basically consists of a simple genetic algorithm with a modified selection mechanism. In each generation, a number of sub-populations are generated by performing selection based on each objective function in turn. As such, for a k-objective problem and a population of size P, k sub-populations of size P/k each are generated. These sub-populations are then shuffled together to obtain a new population of size P, on which the evolutionary operators of crossover and mutation are applied in the conventional manner. VEGA has several problems, of which the most serious is that its selection scheme is opposed to the concept of Pareto dominance. Based on the operations of VEGA, it is likely that a Pareto-optimal solution, which is a good compromise of all the objectives but not the best in any of them, will be discarded.

2) Multi-objective genetic algorithm (MOGA): Fonseca and Fleming (1993) proposed the MOGA with a Pareto ranking scheme that assigns the same smallest

rank value for all non-dominated individuals, while the dominated ones are ranked according to how many individuals in the population are dominating them. MOGA also uses a niche-formation method, which involves computing a similarity threshold σ_{share} for determining the radius of each niche and fitness sharing of solutions within a niche, for diversifying the population. Fonseca and Fleming (1998) extended the domination scheme in MOGA to include goal and priority information for multi-objective optimization. This allows the algorithm to make use of user knowledge, such as preference on certain objective components, optimization constraints, and approximated attainable regions of the Pareto front.

3) Niched Pareto genetic algorithm (NPGA): The salient feature of the NPGA (Horn and Nafpliotis, 1993; Horn *et al*., 1994) is a special tournament selection scheme based on the concepts of Pareto dominance and fitness sharing. Two individuals are chosen at random from the population and they are each compared with a subset of the population. If one is non-dominated and the other is not, the non-dominated one is selected. In the event of a tie, i.e. both are either dominated or non-dominated with respect to the chosen set of individuals, fitness sharing is used to determine the outcome of the tournament.

4) Non-dominated sorting genetic algorithm (NSGA): The basic idea behind NSGA (Srinivas and Deb, 1994) is the ranking process executed before the selection operation. In the ranking procedure, the non-dominated individuals in the population are first identified. These individuals are assumed to constitute the first non-dominated front with a large dummy fitness value. The same fitness value is assigned to all of them. In order to maintain diversity in the population, a sharing method is

then applied. Subsequently, the individuals in the first front are ignored temporarily and the rest of the population is processed in the same way to identify individuals for the second non-dominated front. A dummy fitness value that is kept smaller than the minimum shared dummy fitness of the previous front is assigned to all the individuals belonging to the new front. This process continues until the whole population is classified into non-dominated fronts. A stochastic remainder proportionate selection scheme is then applied to ensure that individuals in the first front have a higher chance of being selected for reproduction than the rest of the population. NSGA has been criticized for its high computational complexity, non-elitist approach, and the need to specify a sharing parameter. These criticisms led to the development of NSGA-II (Deb *et al*., 2002b), which has become one of the most popular Pareto optimization techniques in the multi-objective optimization community.

5) Strength Pareto evolutionary algorithm (SPEA): In SPEA (Zitzler and Thiele, 1999), an archive population is maintained on top of the evolving population. At each generation, the non-dominated individuals in the evolving population are copied to the archive population and any dominated individual in the archive population is removed. If the number of individuals in the archive population exceeds a predefined threshold, the archive population is pruned by means of clustering. Individuals in the archive population are ranked with reference to the members of the evolving population, while individuals in the evolving population are evaluated with reference to the members of the archive population. Fitness sharing is also included in SPEA, where niches are not defined in terms of distance but are based on Pareto

dominance. An improved version of SPEA, named SPEA2, was later developed by Zitzler *et al*. (2001).

6) Pareto archived evolution strategy (PAES): The PAES was proposed as a local search approach for multi-objective optimization of an offline routing problem (Knowles and Corne, 1999) and was later applied to solve a broad range of problems (Knowles and Corne, 2000). The algorithm uses a (1 + 1) evolution strategy, where each parent generates one offspring through mutation. Like SPEA, an archive population is maintained to collect non-dominated solutions. For diversity, the algorithm generates a grid overlaid on the search space and counts the number of solutions in each grid to evaluate how crowded the region that each solution lies in is. A candidate solution is discarded if it is dominated by the incumbent solution or any solution in the archive population. On the other hand, the candidate solution is added to the archive population and replaces the incumbent solution if it dominates the incumbent solution. In the final case, where the candidate and incumbent solutions are incomparable, the decisions of which solution to be the next incumbent solution and whether to include the candidate solution in the archive population are made based on the crowding mechanism. The (1 + 1)-PAES was later generalized to the (μ + λ)-PAES with μ incumbent solutions and λ offspring (Knowles and Corne, 2000).

The algorithms described above are just some of the more representative MOEAs in the literature. Other MOEA-based approaches available in the literature include non-generational evolutionary algorithm (Valenzuela-Rendón and Uresti-Charre, 1997), multi-objective messy genetic algorithm (MOMGA) I and II (Van Veldhuizen

and Lamont, 2000), Pareto envelope-based selection algorithm (PESA) (Corne *et al*., 2000), incrementing multi-objective evolutionary algorithm (IMOEA) (Tan *et al*., 2001c), micro-genetic algorithm for multi-objective optimization (Coello Coello and Pulido, 2001; Pulido and Coello Coello, 2003), Pareto converging genetic algorithm (PCGA) (Kumar and Rockett, 2002), general multi-objective parallel genetic algorithm (GENMOP) (Keller and Lamont, 2004; Knarr *et al*., 2003), multi-objective hierarchical Bayesian optimization algorithm (mohBOA) (Pelikan *et al*., 2005), ε-multi-objective evolutionary algorithm (ε-MOEA) (Deb *et al*., 2005), fast Pareto genetic algorithm (FastPGA) (Eskandari *et al*., 2007), and omni-optimizer (OmniOpt) (Deb and Tiwari, 2008).

2.3 Summary

Despite the state-of-the-art MOEAs that have been reviewed in this chapter, the application of MOEAs to scheduling problems is not that straightforward. Many of these algorithms cannot operate directly on combinatorial problems. The exhaustive analysis of these algorithms accomplished in the literature mostly concentrates on benchmark test problems, whose optimal solutions are known or can be computed exactly. These problems usually come with relatively well-structured solution spaces that have friendly neighborhood compared to combinatorial problems. Many existing evolutionary operators are designed for conventional representations that are geared towards solving the benchmark test problems and are not suitable for scheduling problems. As such, there is a need to investigate the frameworks of these state-of-the-

art MOEAs and make necessary modifications to them before they can be applied to solve scheduling problems.

Chapter 3

The Multi-Objective Evolutionary Algorithm Framework

This chapter details the framework of the multi-objective evolutionary algorithm (MOEA) designed to solve the scheduling problems studied in this thesis. The algorithmic flow of the MOEA is shown in Fig. 3.1. The discussions in this chapter will place emphasis on the problem-independent components of the algorithm, while problem-specific features will be highlighted in the respective chapters.

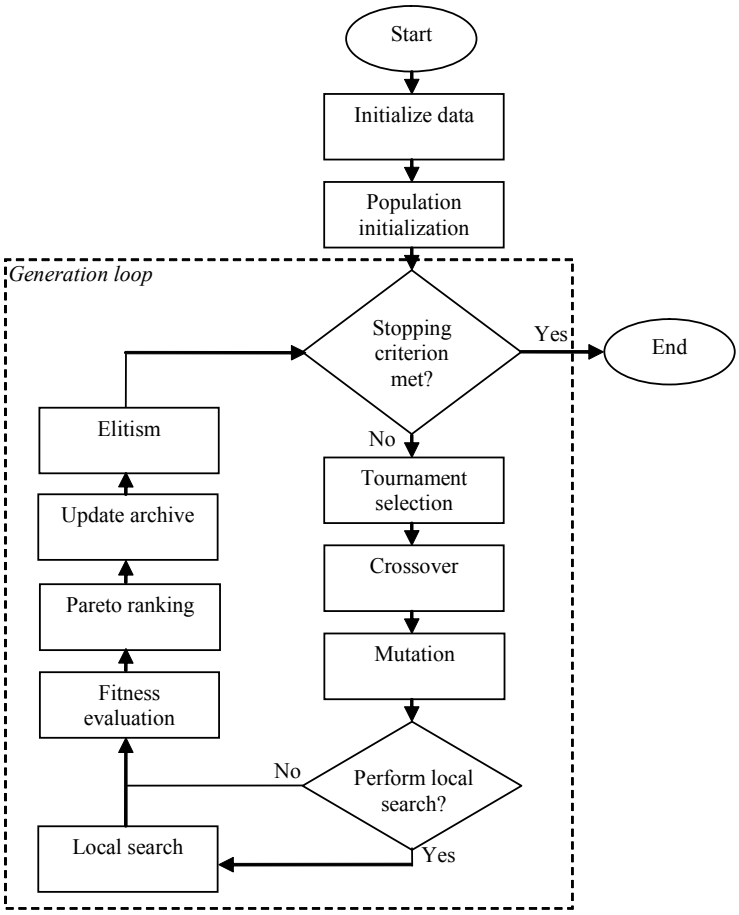


Fig. 3.1 Flowchart of MOEA

3.1 Solution Representation

As defined in the introduction, scheduling is a decision making process which involves the allocation of limited resources to tasks over time. As such, resource and task are two entitles that have to be represented in the chosen solution representation. Unlike conventional EAs, which use a string representation, the MOEA uses a two-dimensional representation (Fig. 3.2), where the columns represent the resources and the rows represent the tasks allocated to each resource. If the number of resources is

fixed, it is referred to as a fixed-length chromosome, otherwise it is referred to as a variable-length chromosome. Both representations are used in the MOEA depending on the problem to be solved. It is also to be noted that the number of tasks allocated to every resource does not need to be the same.

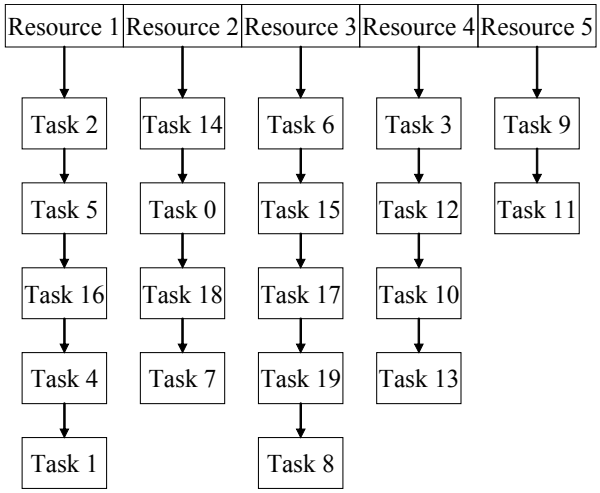


Fig. 3.2 Two-dimensional representation used in MOEA

3.2 Initialization

At the start of the program in Fig. 3.1, problem-specific data is loaded. After which, a population of chromosomes is initialized. The population initialization process involves the use of some problem-specific heuristics, coupled with a stochastic element, to ensure that the initial population covers the more promising areas of the search space evenly.

3.3 Evaluation and Archiving

After the initial evolving population is formed, all the chromosomes are evaluated based on the objective functions and ranked using the Pareto ranking scheme (Fonseca and Fleming, 1993), which assigns the same smallest rank value for all non-dominated chromosomes, while the dominated ones are ranked according to how many chromosomes in the population are dominating them. In Fig. 3.3, a population of seven hypothetical solutions, obtained for a two-objective minimization problem, is plotted in the objective domain. Each solution defines a rectangular box encompassing the origin as shown in the figure. Based on the principle of Pareto dominance defined in Section 2.1.1, for each solution, another solution will dominate the solution if and only if it is within or on the box defined by the first solution but not equal to the first solution in terms of the two objectives. The rank of each of the solutions is also shown in the figure. The rank of a solution is given by $(1 + q)$, where q is the number of solutions in the population dominating the solution. For a three-objective problem, the above explanation still applies, except that each solution will now define a three-dimensional box encompassing the origin instead of a two-dimensional rectangle.

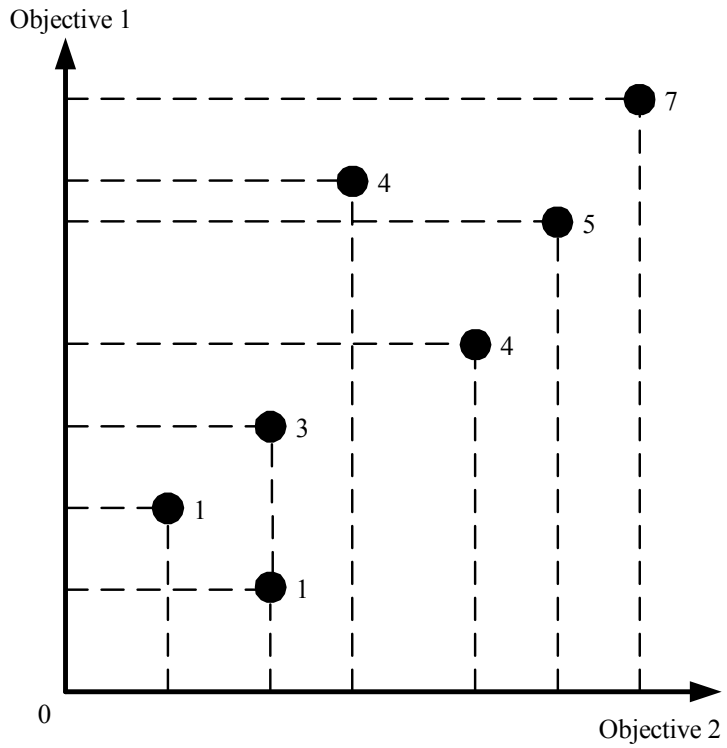


Fig. 3.3 Example to demonstrate Pareto ranking scheme

Following the ranking process, an archive population is updated. The archive population has the same size as the evolving population and is used to store all the best solutions found during the search. The archive population updating process consists of a few steps. The evolving population is first appended to the archive population. All repeated chromosomes, in terms of the objective domain, are deleted. Pareto ranking is then performed on the remaining chromosomes in the population. The higher ranked (weaker) chromosomes are then deleted such that the size of the archive population remains the same as before the updating process. The evolving population remains intact throughout the updating process.

3.4 Genetic Operations

The binary tournament selection scheme is utilized in the MOEA. All the chromosomes in the evolving population are randomly grouped into pairs and from each pair, the chromosome with the lower rank is selected for reproduction. This procedure is performed twice to preserve the original population size. The genetic operators of crossover and mutation are then applied. To further improve the quality of solutions, problem-specific local search operators are applied to the evolving and archive populations at regular intervals for better local exploitation in the evolutionary search.

3.5 Elitism

A simple elitism mechanism is employed in the MOEA for faster convergence. The elitism strategy involves randomly picking a number of non-dominated solutions (5% of the population size) from the archive population. The chosen solutions then replace the worst ranked solutions in the evolving population.

3.6 Stopping Criterion

The operations described in Sections 3.3, 3.4, and 3.5 represent one complete generation of the MOEA and the evolution process iterates until the stopping criterion

is satisfied. Unless otherwise stated, the stopping criterion dictates that the evolution process terminates after a predefined number of generations.

3.7 Summary

This chapter presented the framework of the multi-objective evolutionary algorithm (MOEA) designed to solve the scheduling problems that will be studied in the subsequent chapters. A description of the problem-independent components of the algorithm has been given. Problem-specific operators are designed to adapt this general framework to solve the different scheduling problems.

Chapter 4

Multi-Objective Optimization in Examination Timetabling – A More General Approach

---

This chapter considers the application of the multi-objective evolutionary algorithm (MOEA) described in the previous chapter on a two-objective exam timetabling problem (ETTP). The ETTP involves the scheduling of exams for a set of university courses into a timetable such that there are as few occurrences of students having to take exams in consecutive periods as possible but at the same time minimizing the timetable length and satisfying hard constraints such as limited seating capacity and no overlapping exams. While existing approaches require prior knowledge of the timetable length in order to be effective, the MOEA proposed in this chapter provides a more general solver to the ETTP by including the timetable length as an optimization objective.

4.1 Introduction

The exam timetabling problem (ETTP) is a widely studied combinatorial optimization problem that commonly arises in universities. In recent years, the problem has been getting increasingly difficult as universities are enrolling more students into a wider variety of courses including an increasing number of combined degree courses (Merlot *et al*., 2003). The basic problem involves the allocation of a set of exams to a number of periods (or time slots) so as to satisfy a set of constraints. It follows that different universities have differing views on what constitutes a good exam timetable. This has led to many different formulations of the problem considering different sets of constraints (Burke *et al*., 1996b; Carter and Laporte, 1996; Qu *et al*., 2009; Schaerf, 1999). However, there are two constraints that are universal to all timetabling problems (Burke *et al*., 1996a; Chan *et al*., 2002):

- No student is to be scheduled to take more than one exam at any one time. (Violation of this constraint is referred to as a conflict.)
- For each period, there must be sufficient seats for all the exams that are scheduled for that period.

Due to the criticality of these two constraints, they are usually taken as hard constraints which a timetable must satisfy (at all costs) in order to be feasible. On the other hand, the other constraints are usually taken as soft constraints which are regarded as desirable but not absolutely essential to satisfy all of them. These constraints (Burke *et al*., 1996b) include:

- No student should have to take more than one exam in consecutive periods.

- No student should have to take more than one exam on the same day.
- Large exams should be held earlier during the exam period to allow enough time for grading of the scripts.
- Some exams can only be held in a limited number of periods.
- All exams should be scheduled in less than a particular number of periods.

Quality measures (or objectives) of an exam timetable are usually derived from these soft constraints.

This chapter considers an instance of the ETTP that was first formulated by Burke *et al*. (1996a) but has since received much attention from researchers (Abdullah *et al*., 2007a, 2007b; Caramia *et al*., 2001; Di Gaspero and Schaerf, 2001; Merlot *et al*., 2003; Wong *et al*., 2004). On top of considering the two mentioned universal hard constraints, the problem involves the minimization of the violation of a soft constraint that if a student is scheduled to take two exams in any one day, there should be a free period between the two exams. Violation of this constraint will be referred to as a clash. This constraint is considered with the aim of spreading out the exams for students and allowing them enough time to recover between exams. More details of the problem will be given in the problem formulation in Section 4.2.1.

In minimizing the number of clashes in an exam timetable, Burke and Newall (1999) commented that if a large number of periods were allocated, it would most likely be the case that the clashes can be eliminated. Burke *et al*. (1995) also mentioned that longer timetables are usually required to reduce the number of clashes and that a cap has to be imposed on the number of periods that can be used, otherwise every other period would be empty. From these two observations, it is clear that the

ETTP is inherently a multi-objective optimization problem. In minimizing the number of clashes in an exam timetable, an algorithm for the ETTP must also ensure that the number of periods used is not exceedingly large. Therefore, it is required to minimize multiple conflicting cost functions, such as the number of clashes and the timetable length, concurrently, which is best solved by means of multi-objective optimization. Most of the existing literature, however, use single-objective-based heuristic methods that fix the number of periods that a timetable can use (Abdullah *et al*., 2007a, 2007b; Burke *et al*., 1996a; Caramia *et al*., 2001; Di Gaspero and Schaerf, 2001; Merlot *et al*., 2003). To the authors' knowledge, only Wong *et al*. (2004) has attempted a multi-objective approach to the ETTP instance that is being considered in this chapter. Even then, their approach, which is based on a hybrid multi-objective evolutionary algorithm, utilizes a population that is divided into partitions, each of which contains timetables of a particular length. During the evolutionary process, the lengths of the timetables remain constant. The approach is equivalent to multiple executions of the optimization process, each time using a population with a different timetable length. The approach and many others also require prior knowledge of the timetable length (Abdullah *et al*., 2007a, 2007b; Burke *et al*., 1996a; Caramia *et al*., 2001; Di Gaspero and Schaerf, 2001; Merlot *et al*., 2003). While it has to be acknowledged that universities traditionally know the approximate duration over which the whole examination procedure spans, resulting in most of the existing ETTP research to focus on fixed-length timetables, this approach is hardly optimal from an operational research point of view. Given that the number of students and their course preferences vary for each intake, it is unacceptable that the same timetable length be

used for scheduling exams every year. As such, it is believed that a general algorithm for the ETTP should be able to generate feasible timetables even without presetting the timetable length, especially when a new instance of the problem is first encountered and probably only a range of desired timetable lengths is provided by the timetable planner.

In solving the ETTP, the multi-objective evolutionary algorithm (MOEA) framework described in Chapter 3 is used. The MOEA incorporates two local search operators, namely a micro-genetic algorithm (MGA) and a hill-climber, for local exploitation in the evolutionary search. The algorithm also employs an intuitive variable-length chromosome representation that allows the timetable length to be manipulated during the evolutionary process. In contrast to existing single-objective-based approaches, the MOEA utilizes a goal-based Pareto ranking scheme to solve the multi-objective ETTP. In addition, the algorithm imports several features from the research on the graph coloring problem.

The developed MOEA is tested against a few influential and recent optimization techniques on the Toronto benchmarks (Carter *et al*., 1996) and on the Nottingham instance (Burke *et al*., 1996a), which are the most widely studied datasets in the exam timetabling community. The participating algorithms include Burke *et al*. (1996a), Caramia *et al*. (2001), Di Gaspero and Schaerf (2001), Merlot *et al*. (2003), Wong *et al*. (2004), and Abdullah *et al*. (2007a, 2007b).

This chapter is organized as follows: Section 4.2 gives a brief description of the current state of research on the ETTP as well as the problem formulation of the ETTP instance that is being considered in this chapter. Section 4.3 presents the problem-

specific features of the MOEA designed for solving the ETTP. Section 4.4 presents extensive simulation results and analysis of the proposed algorithm. Conclusions are drawn in Section 4.5.

4.2 Background Information

4.2.1 Problem Formulation

As mentioned in the previous section, this chapter considers an instance of the ETTP that was first formulated by Burke *et al*. (1996a). In this problem, a set of exams $\mathrm{E} = \{e_1, e_2, ..., e_{|\mathrm{E}|}\}$ is to be scheduled into a set of periods $\mathrm{P} = \{1, 2, ..., |\mathrm{P}|\}$, with each period having a seating capacity *S*. There are three periods per weekday and a Saturday morning period. No exam is held on Sundays. It is assumed that the exam period starts on a Monday.

The problem can be formally specified by first defining the following:

- a_{ip} is one if exam e_i is allocated to period *p*, zero otherwise.
- c_{ij} is the number of students registered for exams e_i and e_j.
- s_i is the number of students registered for exam e_i.

The corresponding mathematical formulation is as follows:

Minimize

$$\sum_{i=1}^{|\mathrm{E}|-1}\sum_{j=i+1}^{|\mathrm{E}|}\sum_{p=1}^{|\mathrm{P}|-1} a_{ip}a_{j(p+1)}c_{ij} \tag{4.1}$$

and

$$|\mathrm{P}| \tag{4.2}$$

subject to

$$\sum_{i=1}^{|\mathrm{E}|-1}\sum_{j=i+1}^{|\mathrm{E}|}\sum_{p=1}^{|\mathrm{P}|} a_{ip}a_{jp}c_{ij} = 0 \tag{4.3}$$

$$\sum_{i=1}^{|\mathrm{E}|} a_{ip}s_i \le S, \forall p \in \mathrm{P} \tag{4.4}$$

$$\sum_{p=1}^{|\mathrm{P}|} a_{ip} = 1, \forall i \in \{1,\ldots,|\mathrm{E}|\} \tag{4.5}$$

(4.1) and (4.2) are the two objectives of minimizing the number of clashes and timetable length, respectively. (4.3) is the constraint that no student is to be scheduled to take two exams at any one time, while (4.4) states a capacity constraint that for each period, there must be sufficient seats for all the exams that are scheduled for that period. These two hard constraints define a feasible timetable. (4.5) indicates that every exam can only be scheduled once in any timetable.

4.2.2 Existing State of Research

The ETTP is an annual or semiannual problem for universities and is widely studied by many operational research and computational intelligence researchers due to its complexity and practicality. A wide range of approaches for solving the problem have been proposed and discussed in the existing literature. These approaches can be divided into the following broad categories (Carter, 1986; Petrovic and Burke, 2004; Qu *et al*., 2009): graph-based sequential techniques, clustering-based techniques, constraint-based techniques, meta-heuristics, multi-criteria techniques, hyper-heuristics, and case-based reasoning techniques.

The ETTP, or timetabling problems in general, without any soft constraint, can be modeled as graph coloring problems (Burke *et al*., 2004a; Carter, 1986). In this model, exams are represented as vertices and conflicts between exams are represented as edges between the vertices (Burke *et al*., 2004a; Carter and Johnson, 2001; de Werra, 1985). By taking each color to represent a period in the timetable, the task is then to color the vertices so that no two adjacent vertices have the same color. Several graph coloring heuristics (Brelaz, 1979; Broder, 1964; Carter *et al*., 1996; Wood, 1968) have been proposed in the literature. These heuristics order the exams in some way, e.g. exams with the largest conflict potential first, and then each exam is assigned to a period in that order. Although these heuristics have been widely employed in exam timetabling, they are seldom used alone but hybridized with other search methods (Asmuni *et al*., 2005; Burke *et al*., 1995, 1998a; Burke and Newall, 1999, 2004; Caramia *et al*., 2001; Carter *et al*., 1996; Di Gaspero and Schaerf, 2001).

This is primarily due to their limitation where early assignments may lead to unavailability of feasible periods for exams left later in the construction process.

Clustering-based techniques divide exams into groups such that the exams within each group satisfy all hard constraints. The groups are then assigned to periods with the aim of minimizing the violation of soft constraints (Balakrishnan *et al*., 1992; Lotfi and Cerveny, 1991; White and Chan, 1979).

In constraint-based techniques, such as constraint logic programming (Hentenryck, 1989) and constraint satisfaction techniques (Brailsford *et al*., 1999), exams are represented as finite-domain variables while periods to which an exam can be assigned to without violating any constraint are represented by the values within the domain of the variable representing the exam. Values (periods) are then sequentially assigned to variables (exams) and when no value can be assigned to a particular variable later in the assignment process, a backtracking procedure enables the reassignment of values until a feasible timetable is constructed. Like graph-based sequential techniques, constraint-based techniques are seldom used on their own since they usually cannot provide high quality solutions (Brailsford *et al*., 1999). They are often employed in hybrid algorithms to find an initial feasible solution whose quality is then improved by other intensive search methods (David, 1998; Duong and Lam, 2004; Merlot *et al*., 2003).

Meta-heuristics form the bulk of some of the most successful techniques that have been applied to the ETTP in the past decade. The MOEA proposed in this chapter as well as the few state-of-the-art approaches used to benchmark the performance of the MOEA belong to this category of exam timetabling solvers. Meta-

heuristics can be further divided into two sub-categories – local search-based and population-based. Local search-based meta-heuristics, which include tabu search (Paquete and Stützle, 2003; White and Xie, 2001; White *et al*., 2004), simulated annealing (Bullnheimer, 1998; Burke *et al*., 2004b; Dowsland, 1996; Duong and Lam, 2004; Thompson and Dowsland, 1996a, 1996b, 1998), variable neighborhood search (Burke *et al*., 2006a; Hansen and Mladenovic, 2001; Mladenovic and Hansen, 1997), great deluge algorithms (Burke and Newall, 2003; Burke *et al*., 2004b; Yang and Petrovic, 2005), and greedy randomized adaptive search procedures (GRASP) (Casey and Thompson, 2003), involve searching from an incumbent solution to its neighborhood and are distinguished by their neighborhood structures and moving strategies. Caramia *et al*. (2001), Di Gaspero and Schaerf (2001), Merlot *et al*. (2003), and Abdullah *et al*. (2007a, 2007b) all fall under this sub-category. Caramia *et al*. (2001) developed a local search method based on a set of heuristics. After constructing an initial solution, their algorithm uses a spreading heuristic to reduce the number of clashes while not extending the timetable length. Another heuristic, which extends the timetable by a period and then tries to reduce the number of clashes in the extended timetable, is used if the first one fails to register any improvement. The process is repeated until no further improvement can be found. Di Gaspero and Schaerf (2001) experimented with tabu search. Their tabu search uses a short-term tabu list with random tabu tenure. In the tabu search, two solutions are neighbors if they differ for the period assigned to a single exam. The neighborhood is further reduced by considering only the subset of exams that are involved in constraint violation. To improve the quality of solutions, the algorithm uses the

shifting penalty mechanism of Gendreau *et al*. (1994). Merlot *et al*. (2003) proposed a hybrid algorithm consisting of three phases. In the first phase, an initial solution is built using constraint programming. The quality of the solution is then improved using simulated annealing based on the Kempe chain neighborhood. The last phase involves using a hill-climber to further improve the timetable. Abdullah *et al*. (2007a) adopted a large neighborhood approach based on an improvement graph search methodology originally developed by Ahuja *et al*. (2001) for solving a capacitated minimum cost spanning tree problem. They designed a cyclic-exchange neighborhood that is substantially larger than the traditional two-exchange neighborhood structure. In order to improve computational time, they further developed their algorithm in a later work to store improvement moves in a tabu list (Abdullah *et al*., 2007b). In contrast to local search-based meta-heuristics where a single solution is improved through an iterative process, population-based meta-heuristics, including genetic algorithms (Erben, 2001; Erben and Song, 2005; Ross *et al*., 1996, 1998, 2003; Sheibani, 2003; Terashima-Marin *et al*., 1999a, 1999b), memetic algorithms (Burke *et al*., 1998b; Burke and Newall, 1999; Burke and Landa Silva, 2004; Côté *et al*., 2005), evolution strategies (Gani *et al*., 2004), and ant algorithms (Dowsland and Thompson, 2005; Eley, 2007; Naji Azimi, 2004, 2005), involve the manipulation of a population of solutions in the search space to solve problems. Burke *et al*. (1996a) and Wong *et al*. (2004) belong to this sub-category. Burke *et al*. (1996a) developed a memetic algorithm (Moscato and Norman, 1991; Radcliffe and Surry, 1994) which interleaves the evolutionary operator of mutation with a hill-climber so that the space of possible solutions is reduced to the subspace

of local optima. Wong *et al*. (2004) proposed a hybrid multi-objective evolutionary algorithm. In the algorithm, crossover is replaced by two local search operators. The first operator is designed to repair infeasible timetables produced by the initialization process and the mutation operator. The other local search operator implements a simplified variable neighborhood search meta-heuristic to improve the quality of timetables. An imperfection often associated with meta-heuristics is that they are dependent on parameter tuning and do not work consistently across different ETTP instances. This problem is aggravated by the fact that meta-heuristics are reliant on domain knowledge, i.e. they use a fixed set of heuristics, and are usually tailor made to solve a particular problem.

Multi-criteria or multi-objective techniques are another category of exam timetabling solvers that is very much related to the MOEA proposed in this chapter. As mentioned in Section 4.1, any practical ETTP is usually characterized by a number of soft constraints which define the objectives of the problem. Most existing approaches treat the multi-objective problem as a single-objective one by combining all the objectives via a weighting function. Multi-criteria optimization presents a more general and flexible approach by considering a vector of objectives, which enables all the objectives to be optimized concurrently. Furthermore, it allows a better assessment and understanding of the problem by studying the relationship between the different objectives which are usually conflicting in nature since they are considered from different points of view by different parties involved in the timetabling process (Carter and Laporte, 1996). Despite the suitability of multi-criteria techniques for exam timetabling, there are very few works in the existing

literature that belong to this category (Asmuni *et al*., 2007; Burke *et al*., 2001; Côté *et al*., 2005; Paquete and Fonseca, 2001; Paquete and Stützle, 2003; Petrovic and Bykov, 2003) and only Wong *et al*. (2004) has attempted a multi-criteria approach to the ETTP instance that is being considered in this chapter.

In contrast to the above techniques, hyper-heuristics represent a completely different approach to exam timetabling. Instead of working in a search space of solutions, hyper-heuristics work in a search space of heuristics to select the best set of heuristics for solving the current instance of the problem. This category of exam timetabling solvers (Ahmadi *et al*., 2003; Asmuni *et al*., 2005; Bilgin *et al*., 2007; Burke *et al*., 2005, 2006b, 2007; Hussin, 2005; Kendall and Hussin, 2003, 2005; Qu and Burke, 2005, in press; Ross *et al*., 2004; Terashima-Marin *et al*., 1999c; Yang and Petrovic, 2005) are motivated by the imperfection of meta-heuristics mentioned earlier and are aimed at achieving a higher level of generality.

Case-based reasoning techniques are a relatively recent approach inspired by the human learning process where past experience with a problem is used to solve a newly encountered and similar problem. In terms of exam timetabling, the solutions of previously solved ETTPs are utilized to aid the search of solutions to new problem instances. Such an approach has been employed by Burke *et al*. (2002, 2005, 2006b) and Yang and Petrovic (2005) for exam timetabling.

For the interested readers, there are also a number of comprehensive survey papers on the exam timetabling research in the literature. These include de Werra (1985), Carter (1986), Carter and Laporte (1996), Bardadym (1996), Burke *et al*.

(1996b, 1997), Schaerf (1999), Burke and Petrovic (2002), Petrovic and Burke (2004), and Qu *et al*. (2009).

4.3 Multi-Objective Evolutionary Algorithm

Having seen the framework of the multi-objective evolutionary algorithm (MOEA) in Chapter 3, this section presents several problem-specific features of the MOEA proposed to solve the ETTP by minimizing concurrently the objectives of number of clashes and timetable length.

4.3.1 Variable-Length Chromosome

Most of the existing approaches in the literature use fixed-length timetables. It was mentioned in Section 4.1 that fixed-length timetables inevitably convert the ETTP to a single-objective problem even though it is inherently a multi-objective one. Another problem with fixed-length timetables is that feasibility cannot be guaranteed since it is not always possible to schedule all exams into a fixed-length timetable without violating any of the hard constraints. Special fixing operators have to be designed to ensure that a feasible timetable can be found (Di Gaspero and Schaerf, 2001; Merlot *et al*., 2003; Wong *et al*., 2004).

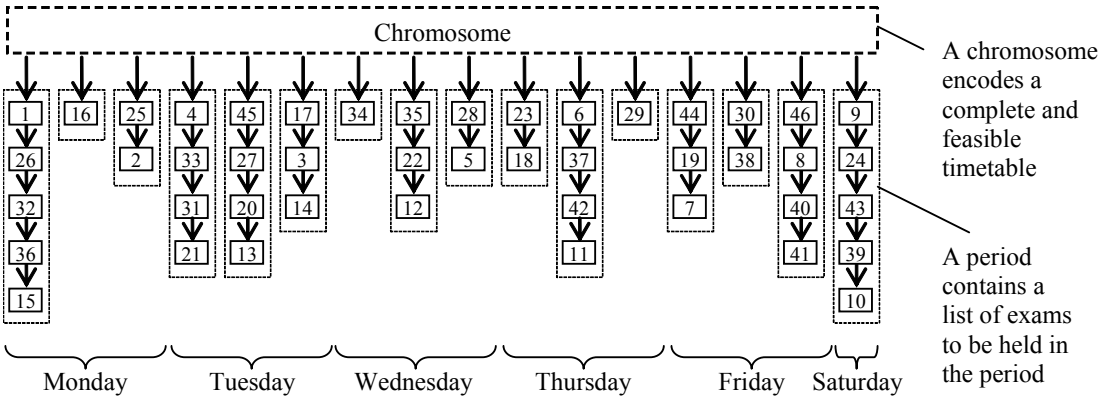


Fig. 4.1 Variable-length chromosome representation

In the MOEA, a variable-length chromosome representation (Tan *et al*., 2003a, 2003b), shown in Fig. 4.1, is applied such that each chromosome encodes a complete and feasible timetable, including the number of periods and the exams scheduled in each of the periods. Such a representation is efficient and allows the number of periods to be manipulated and minimized directly for multi-objective optimization in the ETTP, avoiding the two problems encountered by fixed-length timetables.

4.3.2 Population Initialization

The population initialization process assumes that a desired range of timetable lengths, in the form of maximum and minimum lengths, is provided by the timetable planner. For each chromosome, a timetable with a random number of empty periods within the desired range is created. Exams are then inserted into randomly selected periods of the timetable. The order in which exams are inserted into the timetable is determined by heuristics adopted from the research on the graph coloring problem. It

is widely known that the basic ETTP is a variant of the graph coloring problem. As such, many ETTP researchers have made use of graph coloring heuristics to improve the quality of their timetables (Asmuni *et al*., 2005; Burke *et al*., 1995, 1998a; Burke and Newall, 1999, 2004; Caramia *et al*., 2001; Carter *et al*., 1996; Di Gaspero and Schaerf, 2001). The heuristics used here are based on the belief that if the insertion process concentrates on scheduling those more difficult exams first, it is likely that it would have fewer problems at the end scheduling the easier exams. Five versions of the MOEA based on five different heuristics are tested in this chapter. The heuristics are described below.

1) Largest degree (LD): Exams with the largest number of conflicts with other exams are inserted first.

2) Color degree (CD): Exams with the largest number of conflicts with other exams that have already been scheduled are inserted first.

3) Saturation degree (SD): Exams with the fewest valid periods, in terms of satisfying the hard constraints, remaining in the timetable are inserted first.

4) Extended saturation degree (ESD): Exams with the fewest valid periods, in terms of satisfying both hard and soft constraints, remaining in the timetable are inserted first.

5) Random (RD): Exams are randomly selected for insertion. This is used as a benchmark to check whether the other heuristics are having any effect.

When inserting exams into a timetable, it is very likely that it will come to a point when it is not possible to schedule an exam without violating any of the hard

constraints. In this case, a new period will be created at the end of the timetable to accommodate the exam.

4.3.3 Day-Exchange Crossover

Crossover operators are the way that evolutionary algorithms allow good combinations of genes to be passed between different members of the population. However, most of the existing evolutionary algorithms that have been applied to the ETTP do not use any crossover operator (Burke *et al*., 1996a; Burke and Newall, 1999; Wong *et al*., 2004). Burke and Newall (1999) commented that their experiments with crossover operators for their algorithm have been unfruitful. One criticism that has been leveled against the use of standard crossover operators is that they ignore the notion that "what is good about any timetable is the temporal relationship between exams, rather than their absolute times" (Burke *et al*., 1995). In contrast to standard crossover operators, the day-exchange crossover operator adopted by the MOEA is able to perpetuate favorable temporal relationship between exams. The operation of this crossover is shown in Fig. 4.2.

In day-exchange crossover, only the best days (excluding Saturdays since exams scheduled on Saturdays are always clash-free) of chromosomes, selected based on the crossover rate, are eligible for exchange. The best day consists of three periods and is the day with the lowest number of clashes per student. To ensure the feasibility of chromosomes after the crossover, duplicated exams are deleted. These exams are removed from the original periods while the newly inserted periods are left intact.

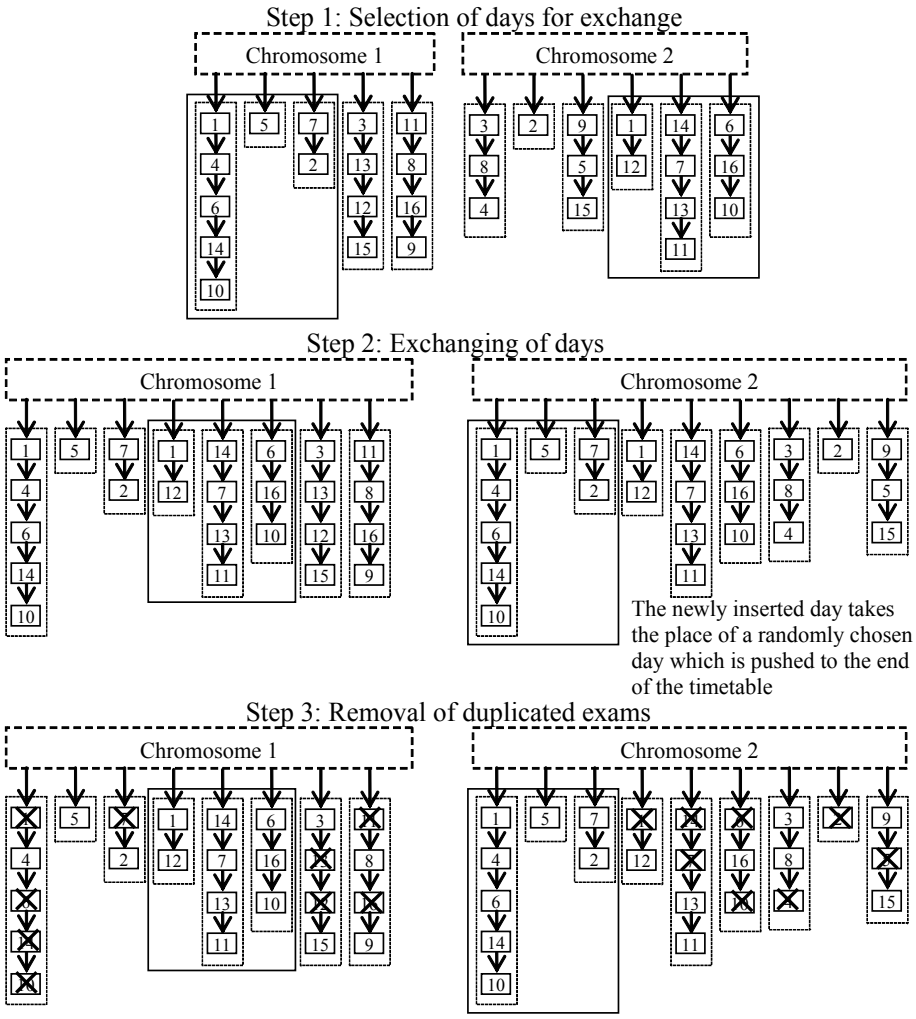


Fig. 4.2 Illustration of day-exchange crossover

From Fig. 4.2, it can be seen that the timetable lengths for the two chromosomes have increased after the crossover operation. In order to control the lengths of timetables after crossover, a period control operator is applied. Chromosomes with timetable lengths within the desired range, which is provided by the timetable planner as mentioned in Section 4.3.2, remain intact, while chromosomes with lengths below the minimum length will undergo a period expansion operation and those with

lengths above the maximum length will undergo a period packing operation. These two operations are described below.

1) Period expansion: The operation first adds empty periods to the end of the timetable such that the timetable length is equal to a random number within the desired range. A clash list, consisting of all exams that are involved in at least one clash, is also maintained. An exam is randomly selected from the clash list and the operation searches in a random order for a period which the selected exam can be rescheduled without causing any clashes while maintaining feasibility. The exam remains intact if no such period exists. The operation ends after one cycle through all exams in the clash list.

2) Period packing: Starting from the period with the smallest number of students, the operation searches in order of available period capacity, starting from the smallest, for a period which can accommodate exams from the former without causing any clashes while maintaining feasibility. The operation stops when it goes one cycle through all periods without rescheduling any exam or when the timetable length is reduced to a random number within the desired range.

4.3.4 Mutation

Mutation operators complement crossover operators in allowing a larger search space to be explored. The MOEA implements a mutation operator that is similar to the light mutation operator of Burke *et al*. (1996a). For each chromosome selected for mutation based on the mutation rate, the operator removes a number of exams,

selected based on the reinsertion rate, from the chromosome. These exams are then reinserted into randomly selected periods while maintaining feasibility. Unlike Burke *et al*. (1996a), the reinsertion process is more elaborate and is based on the graph coloring heuristics used for population initialization as introduced in Section 4.3.2. The order in which exams are reinserted into the timetable is determined by the graph coloring heuristic, depending on the version of the MOEA. Like the population initialization process, when it is not possible to schedule an exam without violating any of the hard constraints, a new period will be created at the end of the timetable to accommodate the exam.

4.3.5 Goal-Based Pareto Ranking

A goal-based Pareto fitness ranking scheme is used in the MOEA to assign the relative strength of solutions. The ranking scheme consists of two phases. The first phase is similar to the Pareto fitness ranking scheme (Fonseca and Fleming, 1993) described in Section 3.3. The second phase of the ranking scheme makes use of the desired range of timetable lengths provided by the timetable planner. The desired range is used as a goal and solutions not meeting the goal are penalized based on the following pseudo-code:

IF *timetable length* > *max length* **THEN**

$rank_2 = rank_1 + (timetable\ length - max\ length)$

ELSE IF *timetable length* < *min length* **THEN**

$rank_2 = rank_1 + (min\ length - timetable\ length)$

$rank_1$ is the rank of a solution after the first phase, whereas $rank_2$ is the adjusted rank after the second phase. The goal-based Pareto ranking scheme allows the MOEA to focus its search on the desired range of timetable lengths and is similar in principle to the goal-sequence domination scheme of Tan *et al*. (2003c).

4.3.6 Local Exploitation

It is widely believed that incorporating local search within evolutionary algorithms is an effective approach for finding high quality exam timetables (Burke *et al*., 1996a; Burke and Newall, 1999; Di Gaspero and Schaerf, 2001; Gani *et al*., 2004; Merlot *et al*., 2003; Wong *et al*., 2004). Local exploitation can contribute to the intensification of the optimization results and is usually regarded as a complement to the evolutionary operators that mainly focus on global exploration. As such, the MOEA utilizes two local search operators, namely a micro-genetic algorithm (MGA) and a hill-climber, which are applied to the evolving and archive populations every 20 generations (setting was chosen after some preliminary experiments). These two operators are applied in turn to chromosomes selected based on a tournament selection scheme, where all the chromosomes in the respective populations are

randomly grouped into fours and from each group, the chromosome with the smallest rank is selected. Only a quarter of each of the populations will undergo local exploitation. Applying local search to a larger proportion of the population has been experimented but no improvement in the results was obtained. A description of the two local search operators is given below.

1) Micro-genetic algorithm: Micro-genetic algorithm (MGA) is a genetic algorithm with small population and short evolution (Coello Coello and Pulido, 2001; Dozier *et al*., 1994; Kazarlis *et al*., 2001; Pulido and Coello Coello, 2003). For each solution produced by the main algorithm that is selected for local search, the operation solves a smaller, single-objective problem by treating each period as an entity and seeks to minimize (4.1) by searching for the optimal order in which the periods are placed in the timetable. The chromosome representation used in MGA is as shown in Fig. 4.3.

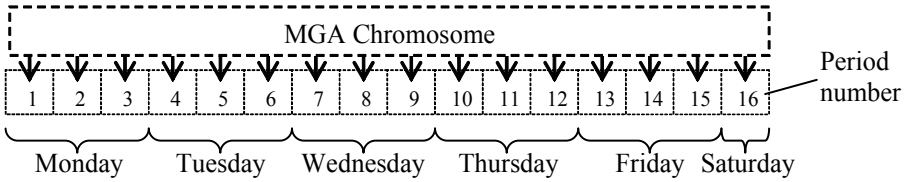


Fig. 4.3 MGA chromosome representation

The main components of MGA are highlighted below:

- *Initialization*: The initial population of MGA is generated by randomly shuffling the order of the periods of the solution provided by the main algorithm.

- *Crossover*: MGA uses an adapted version of the well-known order crossover (Goldberg, 1989; Wang and Zheng, 2003). For each pair of parents, a random fragment of the chromosome from one of them is copied onto the offspring. The empty positions of the offspring are then sequentially filled according to the chromosome of the other parent, following the sequence of periods. The roles of the parents are then reversed to produce the second offspring. The operation is detailed in Fig. 4.4.

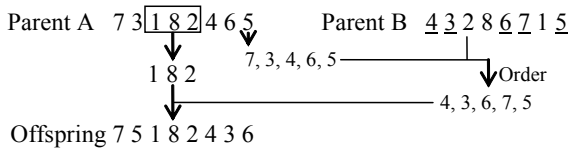


Fig. 4.4 Operation of order crossover

- *Mutation*: Each period will swap position with a randomly chosen period with a probability equal to the swap rate.
- *Selection*: A binary tournament selection scheme is used. All the chromosomes in the MGA population are randomly grouped into pairs and from each pair, the chromosome with the smaller rank is selected for reproduction. This procedure is performed twice to preserve the original population size.
- *Stopping criterion*: MGA stops after a predefined number of generations.

2) Hill-climber: This operation will be applied on the best solution from MGA or the original solution provided by the main algorithm depending on which has a

lower number of clashes. In order to identify the most promising moves, a clash list, like the one used in the period expansion operator, is maintained. Hill-climber operates on a neighborhood defined by randomly selecting an exam from the clash list and rescheduling it in another randomly chosen period or swapping periods with an exam in the chosen period. To avoid the time consuming process of an exhaustive search, only a quarter of the periods will be tested. Hill-climber uses delta evaluation (Burke and Newall, 1999; Ross *et al*., 1994) to avoid performing a full evaluation of each move. The move which leads to the greatest decrease in the number of clashes is selected and the exam is removed from the clash list. If the exam is still not clash-free, it will re-enter the clash list after hill-climber has cycled through all the exams in the clash list. The operation stops when it has cycled through the clash list five times without any improvement in the number of clashes.

4.3.7 Comments on the Desired Range of Timetable Lengths

Although some of the operations of the MOEA require the timetable planner to provide his desired range of timetable lengths, this is not mandatory. Even without the information, the MOEA would still be able to generate feasible timetables by using an arbitrarily large range. It is believed that this is an important feature which a general algorithm for the ETTP should have. In this aspect, the MOEA is superior to most existing single-objective-based approaches which require prior knowledge of the exact timetable length and only produce single-length timetables (Abdullah *et al*., 2007a, 2007b; Burke *et al*., 1996a; Caramia *et al*., 2001; Di Gaspero and Schaerf,

2001; Merlot *et al*., 2003). However, providing the MOEA with the desired range of timetable lengths would allow the algorithm to focus its efforts on the desired range and produce higher quality timetables.

4.4 Simulation Results and Analysis

The MOEA was programmed in C++ and simulations were performed on an Intel Pentium 4 3.2 GHz computer. Table 4.1 shows the parameter settings chosen after some preliminary experiments.

Table 4.1 Parameter settings for simulation study

Parameter	Value
Population size	100
Generation number	200
Crossover rate	0.7
Mutation rate	0.3
Reinsertion rate	0.02
MGA population size	20
MGA generation number	40
MGA crossover rate	0.7
MGA mutation rate	0.3
MGA swap rate	0.3

Carter *et al*. (1996) and Burke *et al*. (1996a) have made several real enrollment datasets for exam timetabling publicly available. Table 4.2 lists the datasets used in this chapter together with the characteristics of each dataset. As all the datasets indicated their desired timetable lengths instead of the desired range of timetable lengths that the MOEA takes as input, a desired range, which includes three periods above and below the indicated desired timetable length, is set for each of the datasets. For example, the desired range for CAR-F-92 is from 37 to 43 periods. It is to be noted that NOT-F-94 indicated two desired timetable lengths. While most single-objective-based approaches would require two separate runs to obtain two timetables with the two desired lengths, the problem can be solved by the MOEA in one run by setting the desired range to be from 23 to 29 periods. It is also important to note that no fine-tuning of the MOEA was performed and the same parameters as shown in Table 4.1 were used in all simulations unless otherwise stated.

Table 4.2 Characteristics of datasets

Dataset code	Number of exams	Number of students	Enrolment	Seating capacity	Number of periods
CAR-F-92	543	18419	55522	2000	40
CAR-S-91	682	16925	56877	1550	51
KFU-S-93	461	5349	25113	1995	20
NOT-F-94	800	7896	33997	1550	23/26
TRE-S-92	261	4360	14901	655	35
UTA-S-92	622	21266	58979	2800	38

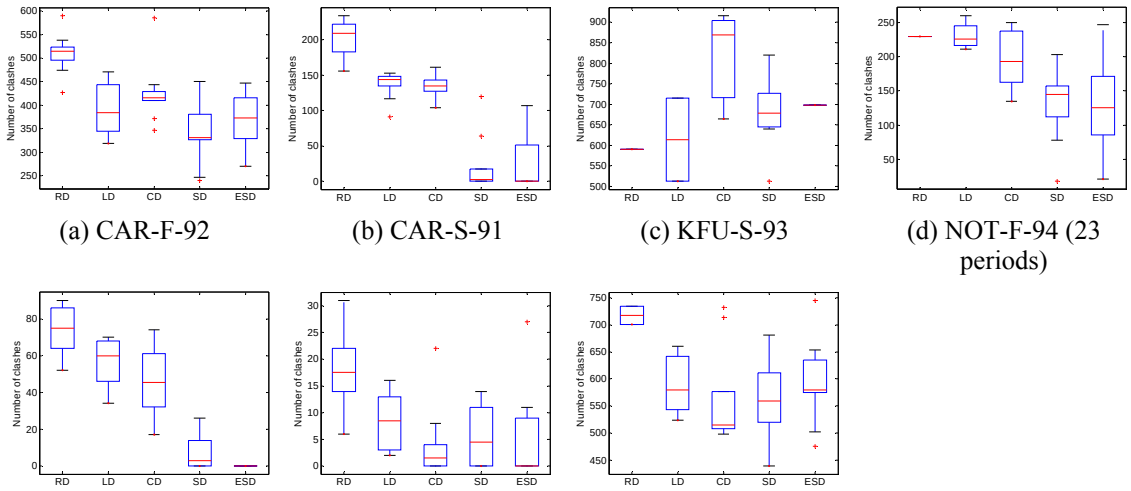
The subsequent sections present the extensive simulation results and analysis of the proposed MOEA. Section 4.4.1 studies the performance of the MOEA based on the different graph coloring heuristics. Sections 4.4.2 and 4.4.3 present, respectively, the contribution of day-exchange crossover and the two local search operators of MGA and hill-climber to the performance of the MOEA. Section 4.4.4 demonstrates the advantages of multi-objective optimization and at the same time validates the relationship between the two objectives of number of clashes and number of periods required in a timetable. Section 4.4.5 shows why the MOEA is a more general ETTP solver compared to existing single-objective-based approaches. Lastly, Section 4.4.6 presents the comparison results of the MOEA with a few influential and recent optimization techniques.

4.4.1 Performance of Graph Coloring Heuristics

Several graph coloring heuristics are incorporated in the MOEA during the solution initialization process as well as in the mutation operator. These heuristics affect the order in which exams are scheduled into the timetable for the two operations and have significant impact on the search trajectory of the MOEA. This section studies the performance of the MOEA based on the different graph coloring heuristics.

The five versions of the MOEA, namely LD, CD, SD, ESD, and RD, using the different graph coloring heuristics described in Section 4.3.2 were applied to the datasets shown in Table 4.2. Ten independent runs of each of the settings on each of the datasets were conducted. The results obtained are represented in box plots and are

shown in Fig. 4.5. Each box plot represents the distribution of the number of clashes for Pareto solutions with the desired number of periods for the 10 runs where the horizontal line within the box encodes the median, and the upper and lower ends of the box are the upper and lower quartiles, respectively. The two horizontal lines beyond the box give an indication of the spread of the data. A plus sign outside the box represents an outlier.



(a) CAR-F-92 (b) CAR-S-91 (c) KFU-S-93 (d) NOT-F-94 (23 periods)

(e) NOT-F-94 (26 periods) (f) TRE-S-92 (g) UTA-S-92

Fig. 4.5 Performance comparison for different graph coloring heuristics

From Fig. 4.5, considering the medians and the variances of the results, it is clear that SD gives the best performance for CAR-F-92, CAR-S-91, and UTA-S-92, while ESD works best on NOT-F-94 (for both desired number of periods) and TRE-S-92. The results for KFU-S-93 are less conclusive since the MOEA, regardless of version, is not able to find solutions with the desired number of periods for some of the runs.

Table 4.3 shows the number of runs that the respective versions of the MOEA are not able to find solutions having the desired number of periods for the various datasets.

Table 4.3 Comparison of number of runs that a solution with the desired timetable length could not be found

	RD	LD	CD	SD	ESD
CAR-F-92	0	0	0	0	0
CAR-S-91	0	0	0	0	0
KFU-S-93	9	8	7	3	9
NOT-F-94 (23)	9	6	3	0	0
NOT-F-94 (26)	0	0	0	0	0
TRE-S-92	0	0	0	0	0
UTA-S-92	8	0	0	0	0

The results in Table 4.3 show that SD is able to find solutions with the desired timetable length for seven out of the 10 runs conducted on KFU-S-93, the most out of the five graph coloring heuristics. It is also obvious that KFU-S-93 is the bane of ESD since the heuristic is only able to produce one timetable with the desired length although its performance is comparable to SD on the other datasets. In general, KFU-S-93 seems to pose some problems to the MOEA, regardless of version. One probable reason for the MOEA's inability to find feasible timetables with the desired length for KFU-S-93 on all the runs could be that the desired number of periods for the dataset is set too low and the number of feasible timetables having the desired length is very small. Another reason could be that since the MOEA is designed to

produce a Pareto set of timetables, its search space is significantly larger than that handled by existing single-objective-based approaches. The MOEA has to spread out its efforts to find timetables with lengths within the desired range instead of focusing only on the desired length. Nonetheless, the MOEA is designed to produce feasible timetables even if it is not able to achieve timetables of the desired length. The five versions of the MOEA are able to schedule all the exams of KFU-S-93 in 21 periods (one period more than desired) for all the simulation runs conducted. This result is a consequence of the use of the variable-length chromosome representation in the MOEA. The representation is flexible as the length of the timetable is not fixed but is allowed to be manipulated during the evolution process. This is unlike most of the existing approaches (Abdullah *et al*., 2007a, 2007b; Burke *et al*., 1996a; Caramia *et al*., 2001; Di Gaspero and Schaerf, 2001; Merlot *et al*., 2003) which fix the timetable length at the desired length and any exam that cannot be inserted into the timetable are left unscheduled. For these approaches, certain operators have to be designed to ensure that all exams are scheduled at the end of the optimization process. Merlot *et al*. (2003) designed a greedy heuristic and relaxed a hard constraint by allowing students to have two exams scheduled at the same time to tackle the case where not all exams are scheduled at the end of the main optimization process. Burke *et al*. (1996a) included in their evaluation function a term to penalize solutions with unscheduled exams. Even with these measures, it is not guaranteed that they will be able to come up with feasible timetables. This problem becomes even more significant when the desired length of timetables is set too low. The MOEA, on the other hand, does not face such a problem. The solutions are kept feasible and all

exams are scheduled throughout the optimization process since the representation allows for the flexibility of increasing the number of periods when the timetable is deemed too short to accommodate all the exams.

Table 4.4 compares the best solutions with the desired timetable lengths obtained by the five graph coloring heuristics for all the datasets. Each grid shows the number of clashes in the solution and the average computation time over the 10 runs performed in brackets. The best solutions for each of the datasets are highlighted in boldface.

Table 4.4 Comparison of best solutions and average computation times (in seconds)

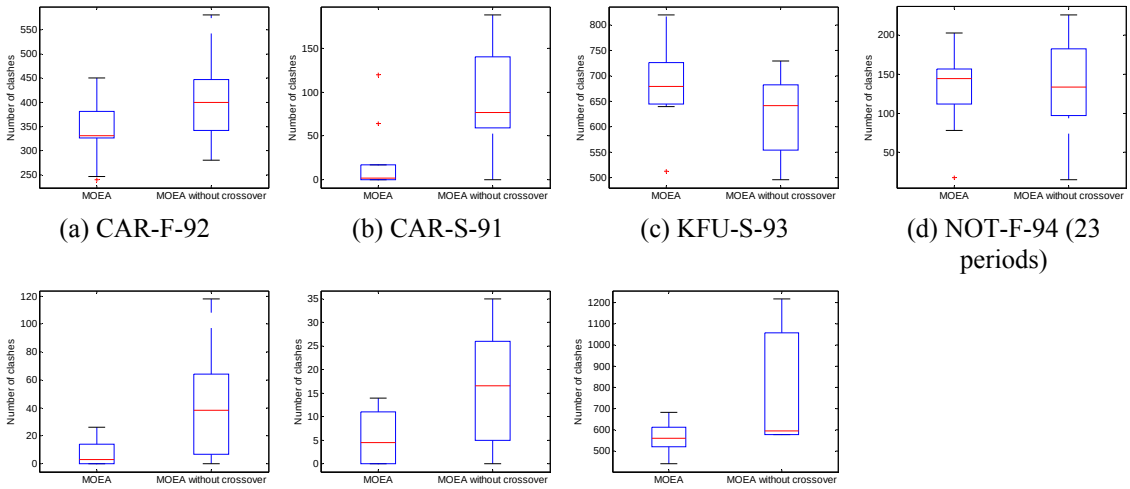
	RD	LD	CD	SD	ESD
CAR-F-92	427 (194.5)	319 (136.7)	347 (142.2)	**240** (172.2)	270 (251.3)
CAR-S-91	156 (141.7)	91 (123.3)	104 (119.7)	**0** (183.3)	**0** (372.3)
KFU-S-93	591 (213.1)	**513** (206.2)	665 (206.9)	**513** (211)	698 (273.6)
NOT-F-94 (23)	230 (217.4)	211 (209)	135 (199.6)	**18** (282.8)	21 (404.5)
NOT-F-94 (26)	52 (193)	34 (184.1)	17 (180.2)	**0** (272.2)	**0** (419.4)
TRE-S-92	6 (30.8)	2 (30)	**0** (30.1)	**0** (36.1)	**0** (50.8)
UTA-S-92	701 (454.4)	524 (294.9)	498 (284.5)	**439** (377.7)	475 (527.1)

From Table 4.4, it is clear that SD dominates over all the other versions of the MOEA in terms of generating the best solutions. The results in this section have shown that the effectiveness of a graph coloring heuristic depends on the structure of the dataset. A heuristic may perform well on some datasets but poorly on others. The results have also shown that graph coloring heuristics can significantly improve the quality of solutions over the random setting. As such, it is beneficial to incorporate some graph coloring heuristics when solving the ETTP but the choice of heuristic is crucial to the success of the algorithm. From the above results, it seems that the saturation degree heuristic is able to perform well in general. On top of being able to find timetables with lower number of clashes, the heuristic is also superior in terms of packing exams into a smaller number of periods. Carter *et al*. (1996), Burke and Newall (1999), and Merlot *et al*. (2003) have also made similar conclusions that the saturation degree heuristic gives the best performance. As such, SD is selected as the default setting for any further analysis of the MOEA unless otherwise stated.

4.4.2 Contribution of Day-Exchange Crossover to the Performance of MOEA

It was mentioned in Section 4.3.3 that most of the existing evolutionary algorithms that have been applied to the ETTP do not use any crossover operator (Burke *et al*., 1996a; Burke and Newall, 1999; Wong *et al*., 2004). The reason is that many researchers find that the inclusion of crossover operators does not bring about any improvement in performance. This section presents the performance improvement that day-exchange crossover brings to the MOEA.

In order to see the effect of day-exchange crossover on the performance of the MOEA, the MOEA was applied to the six datasets without using the operator. The results of this setting based on 10 independent runs are shown in Fig. 4.6. The results of the SD version of the MOEA in Fig. 4.5 have also been included in the plots for comparison. A comparison of the number of runs that the two settings are not able to find solutions having the desired number of periods for the various datasets is shown in Table 4.5. The average computation times over the 10 runs performed are also shown in brackets in Table 4.5.



(a) CAR-F-92 (b) CAR-S-91 (c) KFU-S-93 (d) NOT-F-94 (23 periods)

(e) NOT-F-94 (26 periods) (f) TRE-S-92 (g) UTA-S-92

Fig. 4.6 Performance comparison for MOEA with and without day-exchange crossover

Table 4.5 Comparison of number of runs that a solution with the desired timetable length could not be found and average computation times (in seconds)

	MOEA	MOEA without crossover
CAR-F-92	0 (172.2)	0 (126.7)
CAR-S-91	0 (183.3)	0 (180.7)
KFU-S-93	3 (211)	2 (105.4)
NOT-F-94 (23)	0 (282.8)	2 (262.9)
NOT-F-94 (26)	0 (272.2)	2 (267.4)
TRE-S-92	0 (36.1)	0 (37.1)
UTA-S-92	0 (377.7)	5 (171.2)

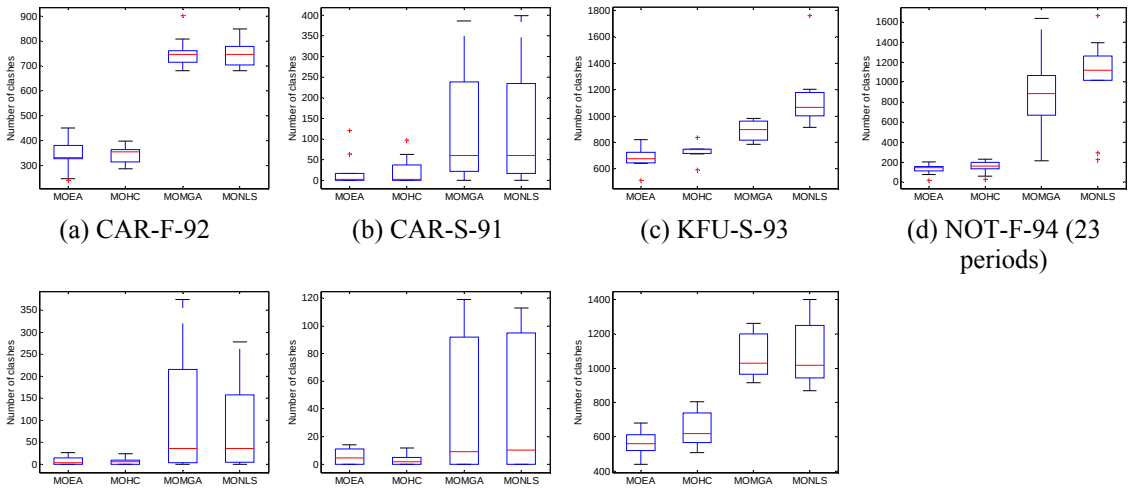
The performance comparison in Fig. 4.6 shows that the MOEA definitely performs better with the crossover operator. With the exception of KFU-S-93 and NOT-F-94 (23 periods), the MOEA, with day-exchange crossover, is able to produce timetables with distinctly lower number of clashes. For NOT-F-94 (23 periods), although the results in Fig. 4.6(d) suggest that the MOEA performs slightly better without the crossover operator, it has to be noted that the setting is not able to find a timetable with the desired number of periods for two of the runs as can be seen in Table 4.5. KFU-S-93 continues to pose a problem for the MOEA. As mentioned, it seems that the relatively poorer performance of the MOEA on the dataset is due to the

dataset's desired number of periods being set too low such that the number of feasible timetables having the desired length is very small. This explanation probably also applies for the slightly poorer performance of the MOEA on NOT-F-94 (23 periods) in Fig. 4.6(d) since the performance of the MOEA is significantly better with day-exchange crossover for NOT-F-94 (26 periods) in Fig. 4.6(e). Table 4.5 shows that with day-exchange crossover, the MOEA is generally more geared towards finding timetables with the desired number of periods.

4.4.3 Contribution of Local Exploitation to the Performance of MOEA

The MOEA incorporates two local search operators, an MGA and a hill-climber, to complement the evolutionary operators of day-exchange crossover and mutation. Like the previous section, this section shows the performance of the MOEA with and without the local search operators.

Simulations were conducted using three other settings. MOHC and MOMGA are the settings which use solely hill-climber and MGA, respectively, for local exploitation. MONLS is the setting that does not use local search at all. Ten independent runs of the three settings are again conducted to obtain statistical results which are shown in Fig. 4.7. The results of the SD version of the MOEA in Fig. 4.5 have again been included in the plots for comparison. The average computation times over the 10 simulation runs performed are shown in Table 4.6.



(a) CAR-F-92 (b) CAR-S-91 (c) KFU-S-93 (d) NOT-F-94 (23 periods)

(e) NOT-F-94 (26 periods) (f) TRE-S-92 (g) UTA-S-92

Fig. 4.7 Performance comparison for MOEA with different local search settings

Table 4.6 Comparison of average computation times (in seconds)

	MOEA	MOHC	MOMGA	MONLS
CAR-F-92	172.2	135.3	147.1	111.8
CAR-S-91	183.3	139.1	160.7	116.7
KFU-S-93	211	168.3	162.7	118.8
NOT-F-94 (23)	282.8	178.7	261.8	157.3
NOT-F-94 (26)	272.2	169.1	251.1	147.6
TRE-S-92	36.1	22.7	33.7	20.4
UTA-S-92	377.7	331.1	312.6	273.2

From Fig. 4.7, the contribution of hill-climber to the performance of the MOEA is obvious since the two settings which use the operator are able to generate solutions with significantly lower number of clashes. In contrast, the effectiveness of MGA is relatively more subtle. It is observed that the inclusion of MGA in the MOEA allows

a slight performance improvement over MOHC for CAR-F-92, CAR-S-91, KFU-S-93, NOT-F-94 (23 periods), and UTA-S-92. It was commented that the desired number of periods for KFU-S-93 and NOT-F-94 (23 periods) have been set too low. The performance improvement attributed to MGA for these two datasets seems to agree well with this comment. For these two datasets, due to the low desired timetable lengths, the timetable would be very tight and the hill-climber will not be able to function to its full potential since the operator requires some allowance to move exams between periods. On the other hand, the operations of MGA, which sought to find the optimal order in which periods are arranged in a timetable, are not affected by how packed the timetable is. Comparing the number of clashes in the best solutions obtained by the MOEA and MOHC in Table 4.7, it is obvious that the inclusion of MGA in the MOEA is vital to the success of the algorithm.

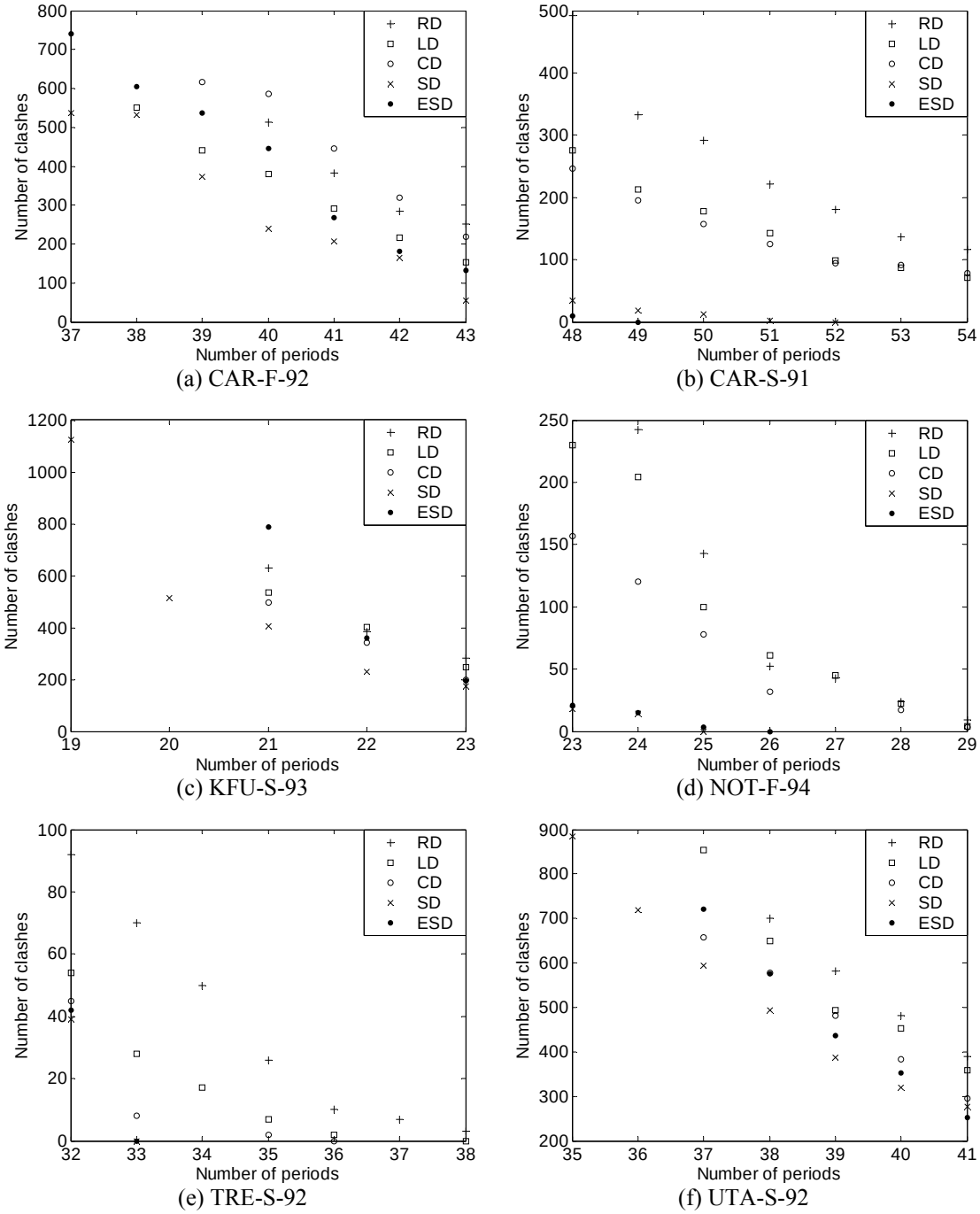
Table 4.7 Comparison of best solutions

	MOEA	MOHC
CAR-F-92	**240**	287
CAR-S-91	**0**	**0**
KFU-S-93	**513**	594
NOT-F-94 (23)	**18**	28
NOT-F-94 (26)	**0**	**0**
TRE-S-92	**0**	**0**
UTA-S-92	**439**	508

4.4.4 Performance of Multi-Objective Optimization

This section presents the multi-objective optimization performance of the MOEA. On top of showing the advantages of multi-objective optimization, the relationship between the two objectives of number of clashes and number of periods required in a timetable will also be validated.

The main role of the MOEA is to generate a Pareto set of timetables from which the timetable planner can make an informed decision. Having seen the results for the desired timetable length in the previous sections, the results for the desired range of timetable lengths for each of the datasets are plotted in Fig. 4.8. The figures show the Pareto set of timetables for a randomly chosen run of each of the five versions of the MOEA on each of the datasets.



(a) CAR-F-92 (b) CAR-S-91

(c) KFU-S-93 (d) NOT-F-94

(e) TRE-S-92 (f) UTA-S-92

Fig. 4.8 Pareto solutions for the datasets

The results in Fig. 4.8 again show that the saturation degree heuristic generally produces lower-clash timetables for all the datasets in comparison to the other graph coloring heuristics.

In addition, the relationship between the two objectives of number of clashes and timetable length can also be observed from Fig. 4.8. It can be seen that the two objectives are conflicting with each other, i.e. any attempt to minimize either of the objectives will cause the other objective to increase. This result shows the importance of taking a multi-objective approach in solving the ETTP. The MOEA is able to minimize concurrently the two conflicting objectives and generate a Pareto set of timetables from which the timetable planner can select a solution to implement based on whether the priority is to have a smaller number of clashes or to conduct the exams in as few periods as possible.

From Fig. 4.8(b), Fig. 4.8(d), and Fig. 4.8(e), it can be observed that clash-free timetables shorter than the desired lengths actually exist. For CAR-S-91, NOT-F-94, and TRE-S-92, the MOEA is able to generate clash-free timetables with 49, 25, and 33 periods, respectively. This is a reduction of up to two periods from the respective desired lengths indicated in Table 4.2. These clash-free results would never have surfaced for existing single-objective-based approaches that only produce single-length timetables.

Experiments were conducted to further examine the multi-objective optimization performance of the MOEA. Two additional types of simulations, with settings similar to the MOEA but have different optimization criteria (for evolutionary selection operation), were performed. The two simulation types are concerned with the single

objectives of minimizing the number of clashes (SOC) and the number of periods (SOP), respectively. Ten independent runs of each of the simulation types were conducted on each of the datasets. The results of this experiment are tabulated in Table 4.8. The table shows the values for the two considered objectives averaged over all the Pareto solutions. It has to be emphasized that, due to their optimization criteria, SOC and SOP produce only one solution each per run. The desired timetable length for each of the datasets is also shown in the table under the respective dataset codes.

Table 4.8 Performance comparison of different optimization criteria

		CAR-F-92 (40)	CAR-S-91 (51)	KFU-S-93 (20)	NOT-F-94 (23/26)	TRE-S-92 (35)	UTA-S-92 (38)
MOEA	Avg. number of periods	40.03	49.96	21.37	25.14	33.76	38.70
	Avg. number of clashes	359.59	59.70	467.56	52.67	25.07	496.52
SOC	Avg. number of periods	48.30	51.90	29.20	27.40	35.56	50.44
	Avg. number of clashes	118.80	0.00	26.00	0.00	0.00	122.78
SOP	Avg. number of periods	35.30	41.10	19.70	22.40	25.90	36.20
	Avg. number of clashes	1774.90	2297.10	719.40	992.80	945.30	780.50

In Table 4.8, SOC and SOP provide two extreme results. The average number of periods of the solutions obtained by SOC for each of the datasets is usually much larger than the corresponding desired number of periods. From the relationship

between the two objectives, it is therefore expected that SOC generates timetables with the lowest number of clashes, which can be seen in Table 4.8. On the other hand, the timetables obtained by SOP are usually much shorter than the corresponding desired number of periods, resulting in them having the largest number of clashes. The MOEA typically produces timetables with lengths around the desired timetable length since the average number of periods of its solutions is relatively closer to the desired timetable length. This leads to its timetables having more moderate number of clashes. To give a visual description of these results, the search spaces in the objective domain explored by a random run of each of the three simulation types on CAR-F-92, which has a desired timetable length of 40, are plotted in Fig. 4.9. Each point in the plots is a point in the objective domain that has been found by the respective simulation types during the operation of the algorithm. The scales of the plots have been kept the same to allow direct comparison of the search spaces.

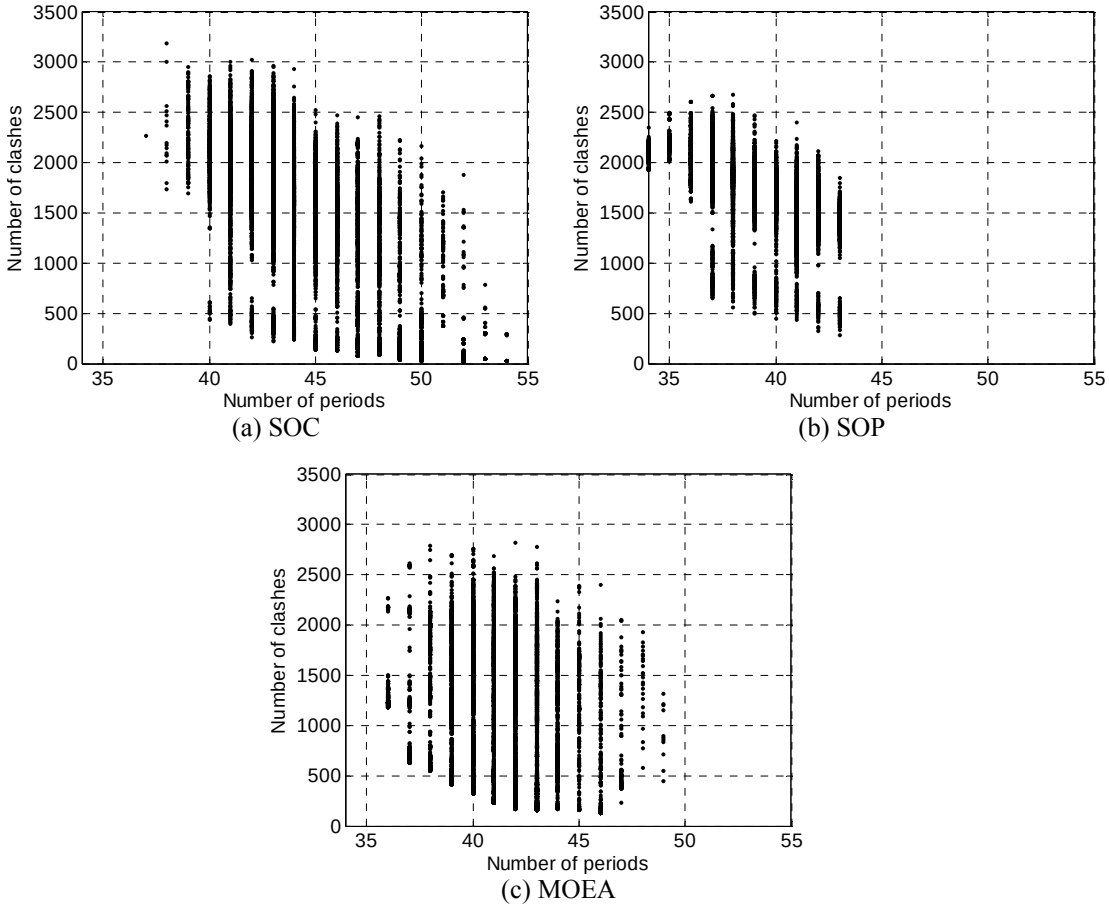


Fig. 4.9 Comparison of search spaces for different optimization criteria

The plots in Fig. 4.9 show that the three simulation types focus their search efforts on different areas of the search space. As can also be seen from the results in Table 4.8, SOC is able to find lower-clash timetables but its search is mainly focused on longer timetables. From the voids in the search space in Fig. 4.9(a), it is clear that very little effort is spent on timetables with lengths around the desired length. From Fig. 4.9(b), SOP concentrates on finding shorter timetables and it is the only simulation type that is able to find feasible timetables with lengths shorter than 36 periods. However, the long and low-clash as well as the short but high-clash

timetables obtained by these two simulation types are definitely sub-optimal as far as the desired timetable length is concerned in this multi-objective optimization problem. Furthermore, they tend to focus their search efforts on a few timetable lengths while neglecting the rest. On the other hand, it can be seen that the MOEA is able to distribute its search efforts to a wider range of periods, focusing particularly on the desired range of periods, which includes three periods above and below the desired timetable length. As such, it can be observed from Fig. 4.9 that, within the desired range of timetable lengths, the solutions obtained by the MOEA are more competitive compared to those obtained by the other two single-objective-based simulation types.

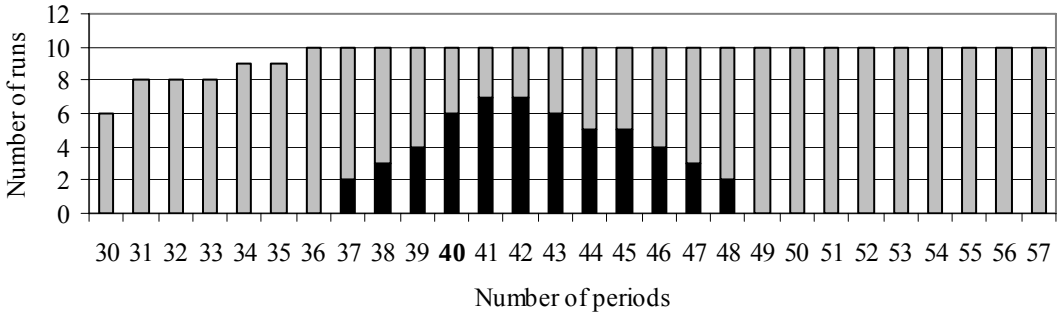
4.4.5 A General Exam Timetabling Problem Solver

The previous section has shown how the MOEA, when provided with information of the desired range of timetable lengths, can focus its search efforts to the desired areas of the search space. This section displays the performance of the MOEA in the absence of period information, i.e. the timetable planner does not provide the desired timetable length or the desired range of timetable lengths.

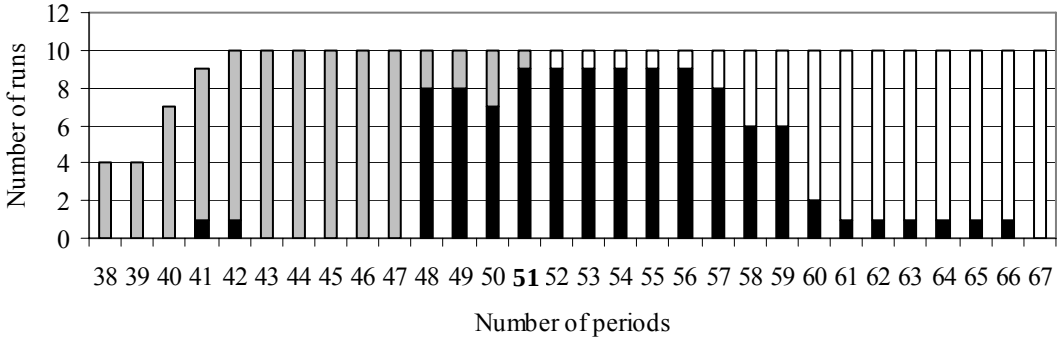
One of the main drawbacks with most of the existing single-objective-based approaches (Abdullah *et al*., 2007a, 2007b; Burke *et al*., 1996a; Caramia *et al*., 2001; Di Gaspero and Schaerf, 2001; Merlot *et al*., 2003) is that they rely strongly on a desired timetable length input from the timetable planner. Even the multi-objective approach taken by Wong *et al*. (2004) required the period information to be effective

in solving the problem. It has been stressed throughout this chapter that a general ETTP solver should be able to generate feasible timetables even without presetting the timetable length. There are a few features of the MOEA that require the timetable planner to provide his desired range of timetable lengths. On top of the goal-based Pareto ranking scheme, the period information is utilized in the population initialization process as well as the period control operator during crossover. Although requiring the timetable planner to provide a desired range of timetable lengths is less demanding compared to requiring a desired timetable length input, it will definitely be more flexible if the MOEA can still perform its task effectively without all these inputs. It has been mentioned in Section 4.3.7 that the MOEA would still be able to generate feasible timetables by using an arbitrarily large range as the desired range. As such, an experiment was conducted using this version of the MOEA, which will be referred to as MONDR, by setting the desired range to be from 1 to 100 periods. MONDR was applied to the datasets and a comparison between the two versions is shown in Fig. 4.10. The plots provide a period-wise comparison of the number of clashes of the Pareto timetables found by the two versions. The normal Pareto ranking scheme (Fonseca and Fleming, 1993) has been used to post-process the timetables found by the two versions to determine the non-dominated timetables so as to include timetables that fall outside the desired range of timetable lengths in the comparison. For simplicity of comparison, the timetables of a run of the MOEA are only compared with their counterparts of the matching MONDR run, i.e. run 1 of MOEA is only compared with run 1 of MONDR. As such, a run-wise, period-wise comparison is made and a point is awarded to the version with the lower number of

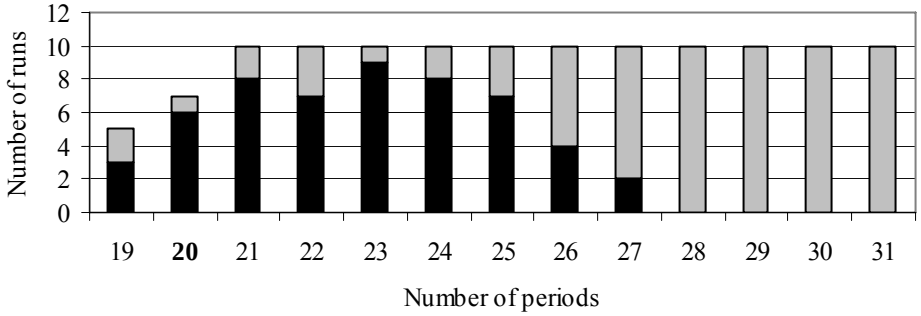
clashes. In the case that both timetables have the same number of clashes, the point goes to ‘equal’. If any of the versions is not represented by a Pareto timetable for any period, i.e. there is a gap in the Pareto front, the timetable with one period shorter is used for the comparison. This is equivalent to adding an imaginary period to that timetable. However, if there is no shorter timetable, an imaginary timetable with an infinitely large number of clashes is used instead. In the case that both versions are represented by this imaginary timetable, no point is awarded. The points obtained by the two versions for each period is accumulated over the 10 runs. From the above description of the comparison system, it can be seen that the total number of points obtained by the two versions and ‘equal’ for a particular period is at most 10. If the total is less than 10, this implies that both versions are not represented by a timetable for that period and they do not have shorter timetables for some of the runs.



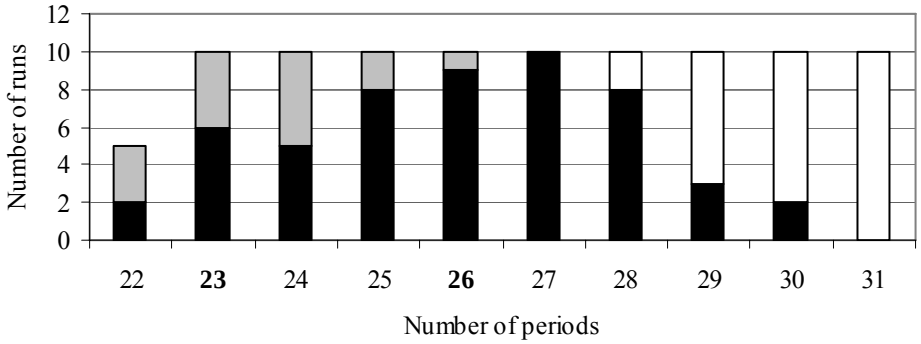
(a) CAR-F-92



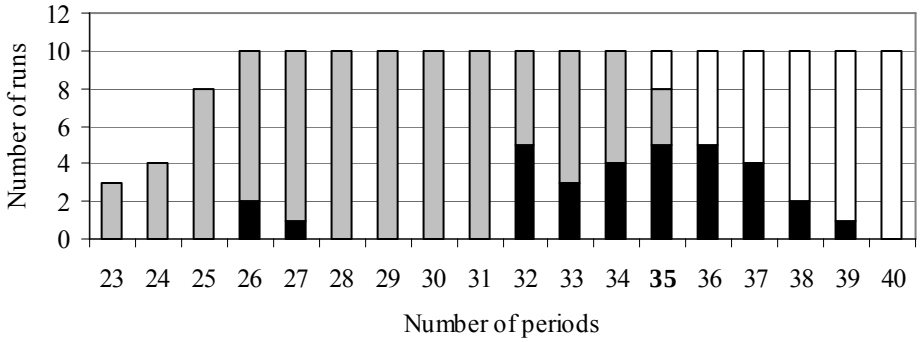
(b) CAR-S-91



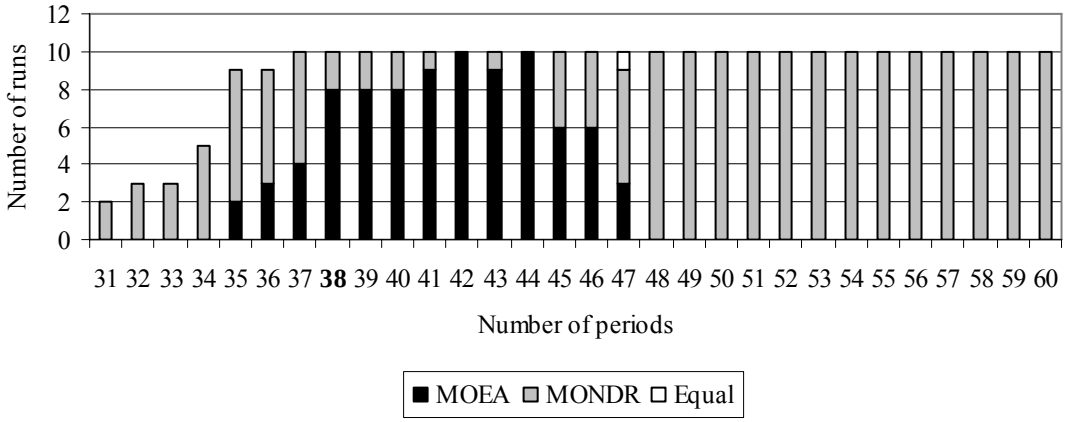
(c) KFU-S-93



(d) NOT-F-94



(e) TRE-S-92



(f) UTA-S-92

Fig. 4.10 Performance comparison of MOEA with and without prior period information

In Fig. 4.10, the black portions of the stacked column charts indicate the points achieved by the MOEA, while the gray areas indicate the points obtained by MONDR in the comparison. The desired timetable lengths for the respective datasets have been highlighted in boldface. The MOEA uses the three periods below and above the desired timetable length as the desired range of timetable lengths for each of the datasets. From the comparison results in Fig. 4.10, it can be observed that the MOEA typically generates lower-clash timetables around the desired range of timetable lengths. Away from the desired range of timetable lengths, MONDR is comparable, if not superior, to the MOEA. The results again show that the three features of the MOEA, which make use of the period information, mentioned at the beginning of this section, can contribute to the intensification of search efforts to the desired range of periods. However, more importantly, the results also show MONDR occasionally coming up with comparable or even better solutions within the desired range of timetable lengths, as well as its emergence for periods away from the desired

range. These results were achieved without prior knowledge of the timetable planner's desired timetable lengths.

To illustrate the scale of the performance difference between the two versions, the Pareto timetables obtained by a random run of MONDR on each of the datasets are shown in Fig. 4.11. The Pareto timetables obtained by the SD version of the MOEA in Fig. 4.8 have also been included in the plots for comparison. The lowest-clash timetables having lengths outside the desired range of timetable lengths have also been included. Although these timetables are not non-dominated under the definition of the goal-based Pareto ranking scheme, they give an indication of the performance of the MOEA outside the desired range of timetable lengths.

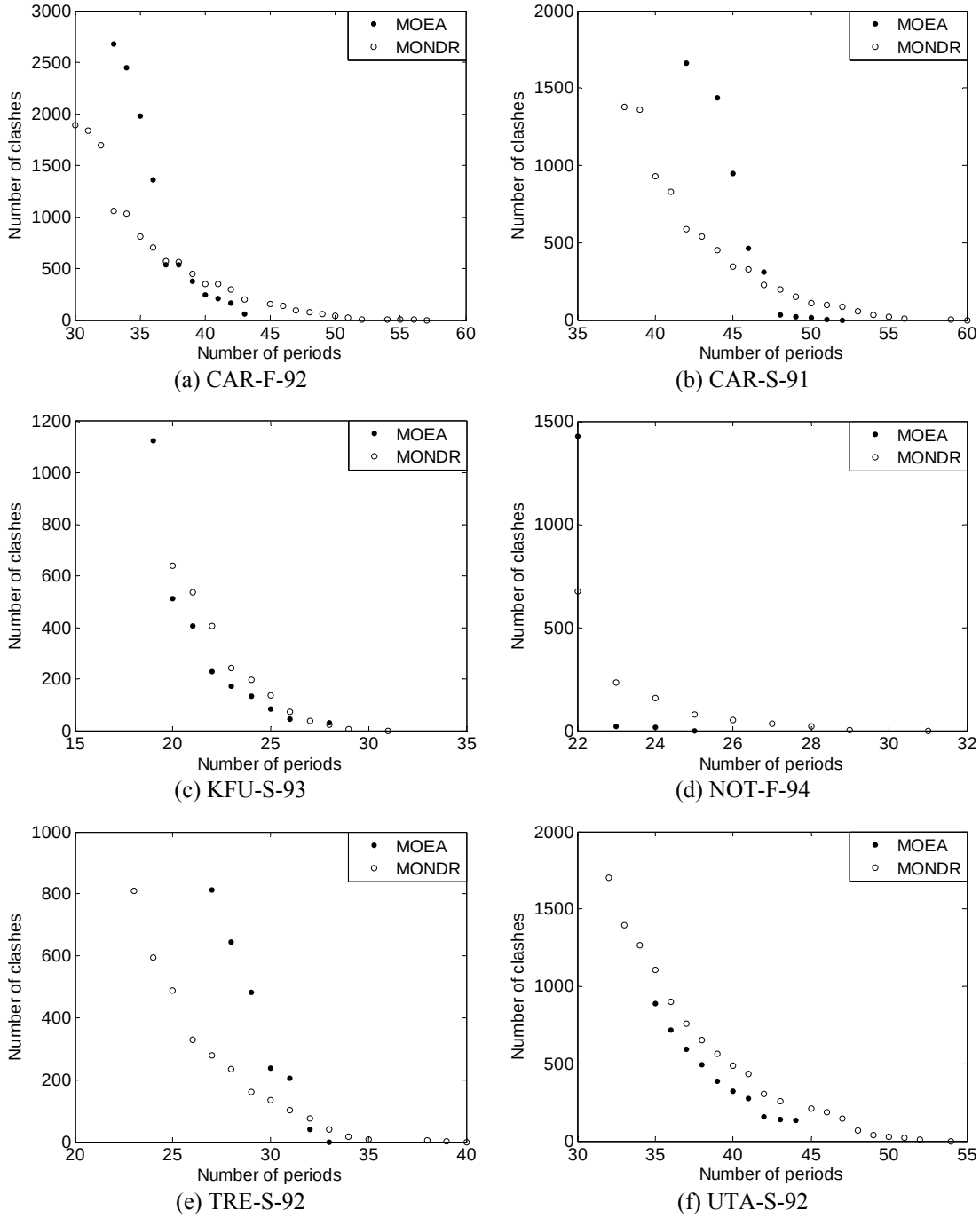


Fig. 4.11 Comparison of Pareto solutions for MOEA and MONDR

From Fig. 4.11, it can be observed that MONDR generally explores a wider range of periods. Due to the lack of period information, MONDR does not

concentrate its search efforts to any range of periods but distribute the efforts to a wider range. The plots in Fig. 4.11 also show that given the understanding that MONDR operates without any guidance of priori information, the quality of the solutions obtained is generally acceptable when benchmarked against those of the MOEA. In contrast, it can be seen from Fig. 4.11(a), Fig. 4.11(b), and Fig. 4.11(e) that the timetables outside the respective desired ranges of periods obtained by the MOEA are definitely inferior to their counterparts generated by MONDR. The results in Fig. 4.10 and Fig. 4.11 are consistent with the 'No free lunch' theorem (Wolpert and Macready, 1995, 1997). While the MOEA outperforms MONDR within the desired range of periods, the opposite occurs outside the range.

To summarize, the results in this section have shown that given prior period information, the MOEA is able to produce lower-clash timetables within the desired range of timetable lengths. The requirement of supplying the MOEA with the desired range of periods to improve the quality of solutions is definitely less demanding than most existing approaches (Abdullah *et al*., 2007a, 2007b; Burke *et al*., 1996a; Caramia *et al*., 2001; Di Gaspero and Schaerf, 2001; Merlot *et al*., 2003), which require the availability of the desired timetable length information since they operate on single-length timetables. While some may argue that these approaches are still able to generate timetables over a desired range of periods through multiple executions of the optimization process, each time setting a different timetable length, this approach is hardly effective. The main problem comes when a timetable planner is not even certain about his desired range of timetable lengths for a newly encountered timetabling problem. Although the timetable planner may face the same

set of courses every year and experience may tell him the desired timetable length or the desired range of timetable lengths to set for the optimization process, the problem evolves over time as the course preference of students change and this can greatly modify the structure of the problem. The fact that the length of a timetable is itself an optimization process further emphasizes the point that the length of a timetable should not be set based on experience. The timetable planner might set his desired range of timetable lengths but a clash-free timetable could actually exist below that range. As such, the importance of a general ETTP solver, which can generate feasible timetables even without any period information, has been emphasized throughout this chapter. In this aspect, this section has shown that the MOEA is still able to produce competitive results by setting it to operate on a large period interval. Of course, the timetable planner could then make use of the results obtained by this setting to decide on his desired range of timetable lengths and then rerun the MOEA based on this range.

4.4.6 Performance Comparison with Established Approaches

To assess the effectiveness of the MOEA, a comparison with a few influential and recent optimization techniques was conducted. Since most of these techniques are based on the single-objective approach, the comparison was carried out using the desired timetable lengths indicated in Table 4.2. The results of the comparison are shown in Table 4.9. In each grid of Table 4.9, there are two numbers representing the number of clashes in the best solution (upper) and the average number of clashes in

solutions (lower). The best solutions for each of the datasets are highlighted in boldface. It has to be noted that there has been some confusion in the literature due to the existence of different datasets having the same name (Qu *et al*., 2009). Efforts have been made to ensure that the results in Table 4.9 were all obtained for the datasets listed in Table 4.2. This is done so that the results obtained by the various optimization techniques can be fairly compared.

Table 4.9 Comparison with other optimization techniques

	MOEA	Burke *et al*. (1996a)	Caramia *et al*. (2001)	Di Gaspero and Schaerf (2001)	Merlot *et al*. (2003)	Wong *et al*. (2004)	Abdullah *et al*. (2007a)	Abdullah *et al*. (2007b)
CAR-F-92	240	331	268	424	**158**	204	525	278
	337.1	-	-	443	212.8	267.4	-	-
CAR-S-91	**0**	81	74	88	31	70	47	37
	21.2	-	-	98	47	78.8	-	-
KFU-S-93	513	974	912	512	247	292	**206**	548
	679.1	-	-	597	282.8	322.9	-	-
NOT-F-94 (23)	**18**	269	-	123	88	156	-	-
	132.1	-	-	134	104.8	182.4	-	-
NOT-F-94 (26)	**0**	53	44	11	2	-	-	18
	7.7	-	-	13	15.6	-	-	-
TRE-S-92	**0**	3	2	4	**0**	**0**	4	**0**
	5.5	-	-	5	0.4	2.4	-	-
UTA-S-92	439	772	680	554	334	**245**	310	300
	561	-	-	625	393.4	338.4	-	-

It can be seen from Table 4.9 that the MOEA produces timetables with the lowest number of clashes for four (CAR-S-91, NOT-F-94 (23 periods), NOT-F-94 (26 periods), and TRE-S-92) out of the seven datasets. The MOEA is ranked third for CAR-F-92 and is ranked fifth for UTA-S-92 and KFU-S-93 albeit falling behind Di Gaspero and Schaerf (2001) in this dataset by only one clash. While some probable reasons explaining why the MOEA is not able to perform as well on some of the datasets have been discussed in Section 4.4.1, it is also widely known that evolutionary algorithms, on which the MOEA is based, produce better results the longer it is allowed to run. In order to test this theory, the MOEA was set to run for 1000 generations, five times longer than it was allowed to run previously, on the three datasets that it could not achieve the best ranking. The results of this experiment are shown in Table 4.10. The average computation times over the 10 runs performed are shown in brackets in Table 4.10.

Table 4.10 Comparison results for long run MOEA and average computation times (in seconds)

	200 Generations	1000 Generations
CAR-F-92	240 337.1 (172.2)	218 286.9 (592.3)
KFU-S-93	513 679.1 (211)	408 617.9 (835.6)
UTA-S-92	439 561 (377.7)	397 514.5 (1391)

From Table 4.10, it is clear that the results get better the longer the MOEA is allowed to run. This characteristic of the MOEA is particularly useful for the ETTP where the time it takes to produce a timetable manually may, in practice, often be measured in months (Burke *et al*., 1996b; Qu *et al*., 2009). While it appears plausible that the MOEA may be able to catch up, in terms of ranking, if it is allowed to perform an even longer run, it is undeniable that the MOEA is not as effective on the three datasets. In spite of this, the MOEA is still proven to be a worthwhile and more general algorithm, among the best that have been applied to the ETTP.

4.5 Summary

This chapter presented an exam timetabling problem (ETTP) which involves the scheduling of exams for a set of university courses. The solution to the ETTP involves the optimization of complete timetables such that there are as few occurrences of students having to take exams in consecutive periods as possible but at the same time minimizing the timetable length and satisfying hard constraints such as limited seating capacity and no overlapping exams. To solve such a multi-objective combinatorial optimization problem, this chapter proposed a multi-objective evolutionary algorithm (MOEA) that uses a variable-length chromosome representation and incorporates a micro-genetic algorithm and a hill-climber for local exploitation and a goal-based Pareto ranking scheme for assigning the relative strength of solutions. It also imports several features from the research on the graph coloring problem. The proposed MOEA has been shown to be a more general exam

timetabling problem solver in that it does not require any prior information of the timetable length to be effective. It has also been tested against a few influential and recent optimization techniques and has been found to be superior on four out of seven publicly available datasets.

Chapter 5

Multi-Objective and Prioritized Berth Allocation in Container Ports

This chapter studies a berth allocation problem (BAP) which requires the determination of exact berthing times and positions of incoming ships in a container port. Unlike the two-objective exam timetabling problem (ETTP) studied in the previous chapter, the BAP considers three objectives of minimizing makespan, waiting time, and degree of deviation from a predetermined priority schedule. These objectives represent the interests of both port and ship operators. The BAP can be considered as an extension to the ETTP in that on top of allocating incoming ships to berths (analogous to allocating exams to periods), the problem requires that the berthing times and positions (within the allocated berth) of the ships to be determined as well. As such, it is essential that the multi-objective evolutionary algorithm (MOEA) is well-adapted to handle this aspect of the BAP.

5.1 Introduction

In many container ports in the US and Japan, berths are leased directly to ship operators where they have exclusive use of the berths. These ports are known as dedicated terminals. The ship operators themselves are in charge of running the operations of the berths. For a ship operator handling large volume of containers and ship calls, the productivity will be high due to economies of scale. However, overcapitalization of the port might result if the handled volume is small as operations will be costly (Imai *et al*., 2001). Multi-user terminals, commonly found in Europe and East Asia, on the other hand are completely run by port operators who will assign incoming ships to any berth, not necessarily the same berth, whenever they call at the port. This type of ports is especially popular in land-scarce countries, such as Singapore and Hong Kong, as they have limited land that can be set aside for berths. The productivity of these ports depends largely on the efficient berth allocation of calling vessels. From the point of view of ship operators, punctuality is an important factor as delays at one port often result in a cascading effect of late port calls at subsequent ports of call for the ship. This can result in heavy losses for shipping companies. Thus, the effective allocation of berths to ships is indeed a very complex and challenging issue that is of concern to both port operators and shipping companies.

This chapter considers the berth allocation problem (BAP) in multi-user terminals. Given a collection of ships that are to arrive at the port within a planning horizon, the BAP involves determining the berthing time and location of each of the

ships, while satisfying a number of spatial and temporal constraints, to optimize operations. More details of the problem are given in Section 5.2.

A number of objectives for optimizing port throughput have been considered in the literature. Imai *et al*. (2001, 2003, 2005, 2007) considered the BAP by minimizing the total service time of ships. The service time of each ship includes the waiting time between the arrival time of the ship at the port and the time the ship berths as well as the handling time for loading or unloading of containers. Guan *et al*. (2002) developed a heuristic for the BAP with the objective of minimizing the total weighted completion time of ship services. Kim and Moon (2003) solved their version of the BAP by minimizing the penalty cost resulting from delays in the departure of ships and additional handling costs resulting from non-optimal locations of ships in the port. According to their formulation, each ship has an optimal berthing location in the port. Park and Kim (2003) solved the same problem by using a sub-gradient method. In their formulation, additional cost is incurred from early or late start of ship handling against their estimated time of arrivals. Li *et al*. (1998) solved the BAP by minimizing the makespan of the schedule. Imai *et al*. (2003) tackled the BAP with service priority, where some ships are given priority, in terms of being serviced earlier, over others. In their work, they provided several examples and arguments for differentiating the service treatment of ships. Lim (1998) took a different approach to the BAP by minimizing the maximum amount of space used for berthing ships. Lai and Shih (1992) proposed some heuristic algorithms for a BAP which is motivated by the need for more efficient berth usage at the HIT terminal of Hong Kong. Their problem assumes the first-come-first-serve (FCFS) allocation strategy which in most

cases does not lead to optimal schedules. Imai *et al*. (1997) considered a BAP for commercial ports. Most service queues are traditionally processed on a FCFS basis. They concluded that for high port throughput, optimal ship-to-berth assignments should be found without considering the FCFS heuristic. However, they also noted that this may result in some dissatisfaction among ship operators regarding the order of the service sequence.

Given the many objectives that have been used to formulate the BAP, it is rather surprising that very little work has been done in the area of multi-objective optimization in BAPs. From the studies of Imai *et al*. (1997), it is apparent that the BAP is inherently a multi-objective optimization problem. An ideal berthing plan for ship operators is one where ships do not have to wait to be berthed and be serviced in the shortest possible time. However, an ideal berthing plan for port operators is one where the makespan, i.e. the time between the first ship that berths at the port and the last ship that leaves, is minimal to achieve full use of their resources at all times. Thus, as port operators try to achieve high throughput in their ports, the satisfaction of ship operators should be considered concurrently. Despite this, most of the existing literature uses single-objective-based heuristic methods that incorporate penalty functions or combine the different objectives by a weighting function (Imai *et al*., 1997, 2003). The drawback of such an objective function approach, as has been discussed in Chapter 1, is that the weights are difficult to be determined precisely, especially when there is insufficient information or knowledge concerning the large real-world BAP. Clearly, these issues can be easily addressed by taking a multi-

objective approach that optimizes all objectives concurrently and effectively without the need of calibrating weighting coefficients.

In this chapter, the multi-objective evolutionary algorithm (MOEA) is applied to solve the BAP by optimizing multiple conflicting objectives from the points of view of port and ship operators. It utilizes the concepts of Pareto optimality to minimize concurrently the makespan of the port and the dissatisfaction of ship operators by reducing the waiting times of their ships. In addition, the MOEA is designed to handle service priority by including the degree of adherence to a predetermined priority schedule as the third objective. This is to allow the port flexibility in giving service priority to ships. Reasons for maintaining a priority system could include terms laid down in shipping contracts, affluence of shipping companies, preference of handling ships with larger or smaller container volume first, or simply the preference of the port management to adopt a FCFS policy to avoid complaints from shipping companies of unfair treatment. To solve this multi-objective optimization problem, the MOEA is equipped with three primary features which are specifically designed to target the optimization of the three objectives. The features include a local search heuristic, a hybrid solution decoding scheme, and an optimal berth insertion procedure. The effects that each of these features has on the quality of berth schedules will be studied.

This chapter is organized as follows: Section 5.2 provides the problem formulation of the BAP. Section 5.3 describes the problem-specific features of the MOEA proposed for solving the multi-objective BAP. Section 5.4 presents extensive

simulation results and analysis of the proposed algorithm. Some concluding remarks are provided in Section 5.5.

5.2 Problem Formulation

The BAP involves allocating a fixed number of berths to a number of ships arriving at the port within the planning horizon for container handling by determining the berthing time and location of each ship. In essence, the BAP bears some resemblance to machine scheduling problems (Guan *et al*., 2002; Li *et al*., 1998), with berths analogous to machines and ships analogous to tasks. However, there are also a number of constraints that are exclusive to the BAP and set it apart from machine scheduling problems.

There are generally two types of berth allocation schemes in the literature. Discrete BAP (Brown *et al*., 1994, 1997; Imai *et al*., 1997, 2001, 2003; Lai and Shih, 1992) considers the berthing space to be a collection of discrete berthing sections where each ship, in terms of length, must fit within the perimeter of its allocated section and only one ship can be serviced in each section at any time. Such a scheme simplifies the problem since it only requires the solver to allocate ships to the finite number of discrete sections. On the other hand, continuous BAP (Guan *et al*., 2002; Kim and Moon, 2003; Li *et al*., 1998; Lim, 1998; Park and Kim, 2002, 2003) considers the berthing space to be a continuous stretch and multiple ships can be berthed simultaneously along the stretch as long as the ships are within the perimeter of the space. This scheme is more complex as it requires the solver to determine the

exact berthing location of each ship along the continuous stretch but it allows the berthing space to be utilized more efficiently. Most of the existing literature that adopted the continuous berth allocation scheme limited their studies to a single continuous berthing stretch where all ships are scheduled within the stretch (Guan *et al*., 2002; Kim and Moon, 2003; Li *et al*., 1998; Lim, 1998; Park and Kim, 2002, 2003). This chapter adopts a more general scheme where the entire berthing space consists of a number of discrete sections and the space is continuous within each section. Each of these discrete sections represents a berth. This scheme is a hybridization of the discrete and continuous BAPs in that the problem involves the allocation of incoming ships to berths and the determination of the exact berthing location of each ship within its allocated berth. Such a scheme is closer to real-world settings where a port consists of a number of berths which are separated geographically. This hybrid scheme has been adopted by Nishimura *et al*. (2001) but their formulation does not require the determination of the exact berthing location of each ship within its allocated berth. In their work, ships are allowed to be serviced simultaneously as long as the sum of their lengths does not exceed the berth length. In reality, this is often not the case. If a ship occupies the centre of a berth leaving empty berth space at the sides, the succeeding ship may not be able to berth within the perimeter of the berth even if the berth length condition is satisfied. Therefore, a more relevant BAP is one where the exact berthing positions of ships are determined. On top of the physical constraint that a ship must be berthed within the perimeter of its allocated berth, each berth also has a water depth and ships with drafts larger than the depth are not allowed to be allocated to the berth.

Regardless of berth allocation scheme, the sequence of events that takes place for each ship calling at the port is the same. Each of the ships will come into the port, wait for the scheduled berthing time, berth at the designated position within the allocated berth, load or unload containers, and leave the port. Fig. 5.1 shows the berth operation timeline for each ship. In Fig. 5.1, the service time of a ship at the port includes the waiting time between the arrival time of the ship and the time the ship berths as well as the handling time for loading or unloading containers.

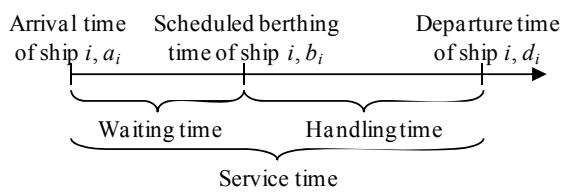


Fig. 5.1 Berth operation timeline

The handling time for each ship is different at different berths. This is to take into account the transportation time for moving the containers to be loaded onto the ship from the original storage area to the allocated berth (Imai *et al*., 2001, 2003, 2005; Nishimura *et al*., 2001). This handling time is also assumed to be deterministic.

The BAP studied in this chapter is then formulated as follows:

Minimize

$$\underset{i \in V}{Max}\,(d_i) - \underset{i \in V}{Min}\,(b_i) \tag{5.1}$$

and

$$\sum_{i \in V} (b_i - a_i) \tag{5.2}$$

and

$$\sum_{i \in V} \frac{(bo_i - po_i) + |bo_i - po_i|}{2} \tag{5.3}$$

subject to

$$\sum_{j \in B} x_{ij} = 1 \quad \forall i \in V \tag{5.4}$$

$$b_i - a_i \geq 0 \quad \forall i \in V \tag{5.5}$$

$$(DB_j - DS_i)x_{ij} \geq 0 \quad \forall i \in V, \forall j \in B \tag{5.6}$$

$$(LB_j - LS_i)x_{ij} \geq 0 \quad \forall i \in V, \forall j \in B \tag{5.7}$$

$$\sum_{j \in B} \left(LB_j - \sum_{i' \in V - \{i\}} LS_{i'} y_{ii'} x_{i'j} - LB_j \right) x_{ij} \geq 0 \quad \forall i \in V \tag{5.8}$$

$$|\, p_{ij} - p_{i'j} \,|\, \delta^p_{ii'j} \geq \frac{LS_i - LS_{i'}}{2} \delta^p_{ii'j} \quad \forall i, i'(\neq i) \in V, \forall j \in B \tag{5.9}$$

$$\left| \frac{b_i + d_i}{2} - \frac{b_{i'} + d_{i'}}{2} \right| \delta^t_{ii'} j \quad \geq \frac{h_{ij} + h_{ij'}}{2} \delta^t_{ii'j} \quad \forall i, i'(\neq i) \in V, \forall j \in B \tag{5.10}$$

$$\delta^p_{ii'j} + \delta^t_{ii'j} = 1 \quad \forall i, i'(\neq i) \in V, \forall j \in B \tag{5.11}$$

$$x_{ij} \in \{0,1\} \quad \forall i \in V, j \in B \tag{5.12}$$

$$p_{ij} \in \mathbb{Z}^+ \quad \forall i \in V, j \in B \tag{5.13}$$

$$b_i \in \mathbb{Z}^+ \quad \forall i \in V \tag{5.14}$$

where B is the set of berths, V is the set of ships, a_i is the arrival time of ship i, b_i is the berthing time of ship i, d_i is the departure time of ship i, h_{ij} is the handling time of ship i at berth j, po_i is the priority order of ship i, bo_i is the berthing order of ship i, DB_j is the water depth of berth j, DS_i is the draft of ship i including the safety vertical distance for berthing, LB_j is the length of berth j, LS_i is the length of ship i including the safety horizontal length, and p_{ij} is the position of ship i in berth j. $x_{ij} = 1$ if ship i is serviced at berth j, $x_{ij} = 0$ otherwise. $y_{ii'} = 1$ if ship i begins its service when ship i' is being serviced at the same berth, $y_{ii'} = 0$ otherwise. $\delta^p_{ii'j} = 1$ if the non-overlapping restriction of berth space in berth j is applied for ships i and i', $\delta^p_{ii'j} = 0$ otherwise. $\delta^t_{ii'j} = 1$ if the non-overlapping restriction of time in berth j is applied for ships i and i', $\delta^t_{ii'j} = 0$ otherwise.

In the problem formulation above, function (5.1) represents the objective of minimizing the makespan of the port. The makespan is defined as the amount of time between the first ship that berths and the last ship that leaves the port. The second objective is represented by function (5.2), which minimizes the total waiting time incurred by ships. Function (5.3) represents the third objective of adhering as closely as possible to a predetermined priority schedule by minimizing the total number of crossings between ships. The berthing order is derived by arranging the scheduled ships, regardless of berth, in order of increasing berthing times. The first ship that

berths is given a berthing order value of 1, while the second ship is given a value of 2 and so on. Similarly, the ship that is given the highest berthing priority is assigned a priority order value of 1, while the next ship is given a value of 2 and so on. The number of crossings contributed by a particular ship is then defined as the difference between its berthing order and priority order when its berthing order is greater than its priority order ($bo_i > po_i$). These three objectives constitute the multi-objective nature of the BAP considered in this chapter. Constraint (5.4) ensures that every ship can only be serviced at one berth without disruption. Constraint (5.5) ensures that ships are serviced only after their arrivals. Constraints (5.6) and (5.7) guarantee that the berths that ships are allocated satisfy the physical properties in terms of berth length and water depth. Constraint (5.8) ensures that the sum of the lengths of ships being serviced simultaneously at a berth does not exceed the length of the berth. Constraints (5.9) and (5.10) are the non-overlapping restrictions. Constraint (5.11) requires that for ships berthed at the same berth, either non-overlapping in berth space or time should be satisfied at all times, i.e. ships allocated to the same berth are not allowed to overlap in terms of both space and time as that would signify a collision. Constraints (5.12), (5.13), and (5.14) show the domains of the three decision variables of the BAP.

5.3 Multi-Objective Evolutionary Algorithm

From the discussions in Section 5.1, it is clear that the BAP is inherently a multi-objective problem. This section presents the multi-objective evolutionary algorithm

(MOEA), focusing on the problem-specific features, designed to solve the BAP by minimizing concurrently the three objectives of makespan of the port, waiting times of ships, and number of crossings.

5.3.1 Fixed-Length Chromosome

A fixed-length chromosome representation (Fig. 5.2) is used in the MOEA. Each chromosome encodes a complete and feasible berth schedule and consists of a fixed number of berths. Each berth consists of a number of ships that are allocated to the berth. The order of ships within each berth indicates the order in which the ships are assigned berthing space and time. This assignment is carried out using two different solution decoding schemes which will be described in the next section. In the figure, ships 2 and 5 are allocated to berth 1 and ship 2 is assigned berthing space and time before ship 5.

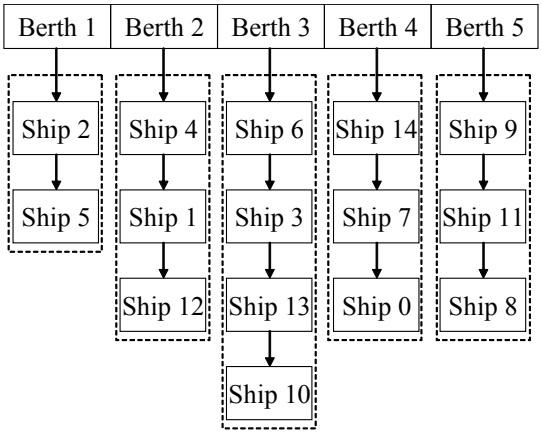


Fig. 5.2 Fixed-length chromosome representation

5.3.2 Solution Decoding

Given the order of ships in each of the berths in a chromosome, the MOEA has to decode the candidate solution by assigning the exact berthing positions and times of the ships (Fig. 5.3). In this way, the departure, waiting, and service times of the ships can be determined, leading to the fitness or objective values of the chromosome.

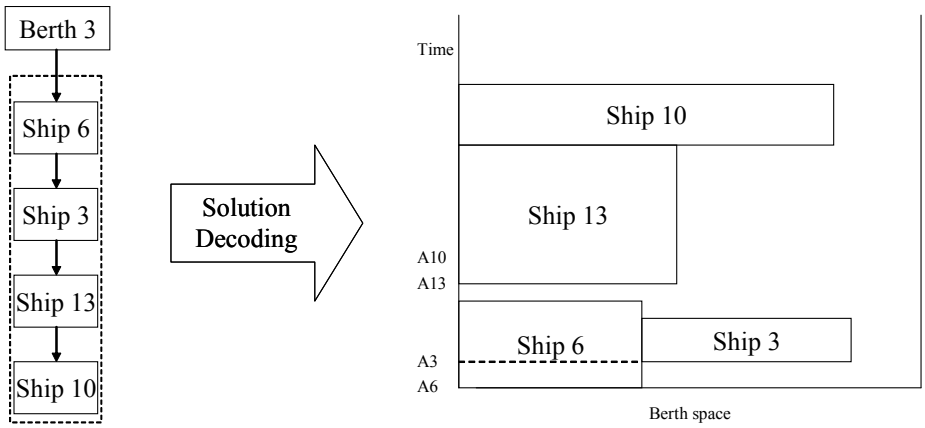
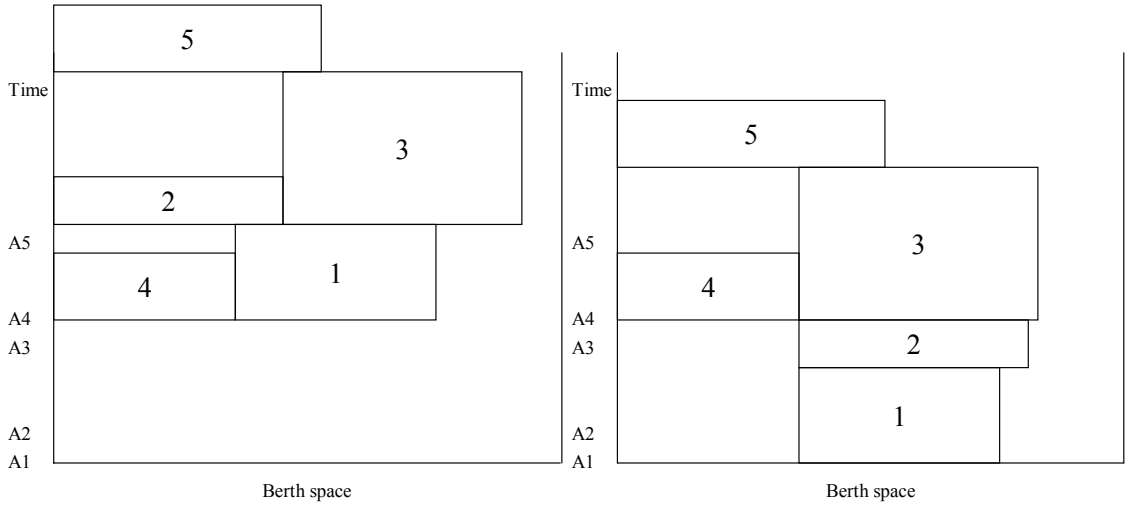


Fig. 5.3 Illustration of solution decoding

From Fig. 5.3, it can be seen that the schedule of a particular berth in the port can be represented by a two-dimensional plane. The horizontal axis represents the position in the berth, while the vertical axis is the time axis. Each ship is represented by a rectangle such that the length of the rectangle is the length of the ship and the height of the rectangle is the handling time of the ship at the berth. The bottom-left corner of the rectangle represents the berthing time of the ship while the top-left corner represents its departure time. A6, A3, A13, and A10 in the figure represent the arrival times of the respective ships.

A simple solution decoding scheme to obtain the berth schedule is to treat the order of ships within each berth of a chromosome as the berthing order, i.e. a ship can only berth at the same time or later than its preceding ship. This scheme will be referred to as the berthing order decoding scheme. An example of how this decoding scheme works is illustrated in Fig. 5.4(a). The order of ships for the particular berth in the chromosome is ship 4 → ship 1 → ship 2 → ship 3 → ship 5. In Fig. 5.4(a), ship 4 is first assigned the leftmost position of the berth as soon as it arrives at A4. Next, due to the solution decoding scheme, ship 1 cannot berth at A1 even though berth space is available. It can only berth at the same time or later than ship 4. Since the berth is long enough to accommodate the simultaneous servicing of ships 1 and 4, ship 1 berths at A4 next to ship 4. On the other hand, ship 2 cannot berth at A4 as the berth is not long enough to accommodate the simultaneous servicing of the three ships. The departure of ship 4 also does not release enough berth space to accommodate ship 2. As such, ship 2 is assigned the leftmost position of the berth after ship 1 has departed from the berth. At the same time, ship 3 berths alongside ship 2 since it has already arrived at A3. Lastly, ship 5 has to wait until ship 3 has left the berth before it gets to berth due to insufficient berth space.



(a) Berthing order 4 1 2 3 5 (b) Assignment order 4 1 2 3 5

Fig. 5.4 Illustration of different solution decoding schemes

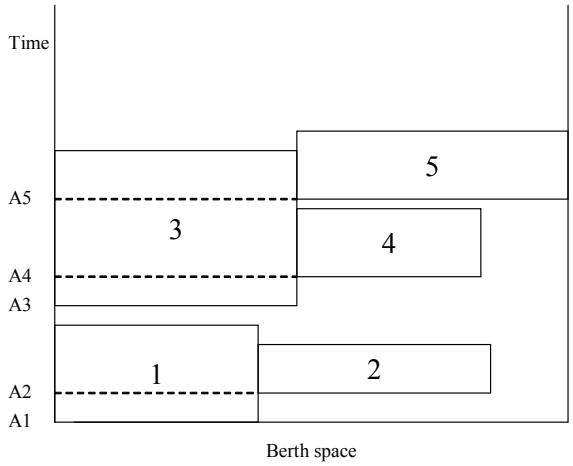
Other than the berthing order decoding scheme, this chapter proposes another decoding scheme which treats the order of ships within each berth of a chromosome as the assignment order, assigning a ship a feasible berth space with the earliest possible berthing time starting from the left. In this scheme, a ship may berth earlier than its preceding ships as long as it has arrived at the port and berth space is available. This scheme will be referred to as the assignment order decoding scheme. An example to show how this decoding scheme works is illustrated in Fig. 5.4(b). The order of ships in the chromosome is the same as that used in the berthing order decoding scheme example. Like the berthing order decoding scheme, ship 4 is assigned the leftmost position of the berth as soon as it arrives at A4. A characteristic of this decoding scheme is that the assignment of berth space to a particular ship renders the berth space unavailable to succeeding ships until the ship has left the berth, i.e. in this case the berth space occupied by ship 4 is made unavailable to ships

1, 2, 3, and 5 until ship 4 has departed from the berth. Unlike the berthing order decoding scheme, ship 1 is allowed to berth at A1. However, the scheme dictates that ships are always assigned the leftmost position of the berth whenever it is available. Since the space to be occupied by ship 4 has already been rendered unavailable, ship 1 berths at the earliest leftmost available berth space, which is the space next to ship 4. Another characteristic of this scheme is that it has two main criteria for determining the berthing location of each ship – earliest possible berthing time and leftmost position, with the former taking precedence over the latter. It is for this reason that ship 1 is not berthed at a position where it has to wait for ship 4 to complete servicing even though that position is to the left of its assigned berthing position. Next, ship 2 is unable to berth beside ship 1 at A2 since the available berth space is not long enough to accommodate the ship. As such, it takes over the berth space from ship 1 after ship 1 has departed from the berth. The same reason explains the assignment of berth space and time to ship 3. With the inclusion of ship 3 in the schedule, the available berth space includes the space previously occupied by ship 4 and the space next to ship 3. However, both spaces are not long enough to accommodate ship 5. As such, ship 5 can only berth after ship 3 has left the berth and it berths at the leftmost position of the berth.

From Fig. 5.4, it can be seen that the berth schedules obtained using the two decoding schemes are very different even though they originated from the same chromosome. This implies that a chromosome may have two different sets of objective values based on the two decoding schemes. The effects of the two solution

decoding schemes on berth schedule quality will be studied and discussed in Section 5.4.2.

As a sidenote, one can observe that the berth schedules in Fig. 5.4 are not favorable as there are voids in the berth schedules resulting in inefficient usage of the berth space. Fig. 5.5 shows the berth schedule decoded from the sequence ship 1 → ship 2 → ship 3 → ship 4 → ship 5. In this case, both solution decoding schemes lead to the same berth schedule. It is obvious that the schedule in Fig. 5.5 has a lower makespan and waiting time than the two schedules in Fig. 5.4. From Fig. 5.4 and Fig. 5.5, it is clear that the order of ships in a chromosome is an important consideration as it will affect how the berth schedule turns out. Inefficient berth schedules will result in unsatisfactory objective values of makespan, waiting time, and number of crossings.



Assignment / berthing order 1 2 3 4 5

Fig. 5.5 A more favorable berth schedule

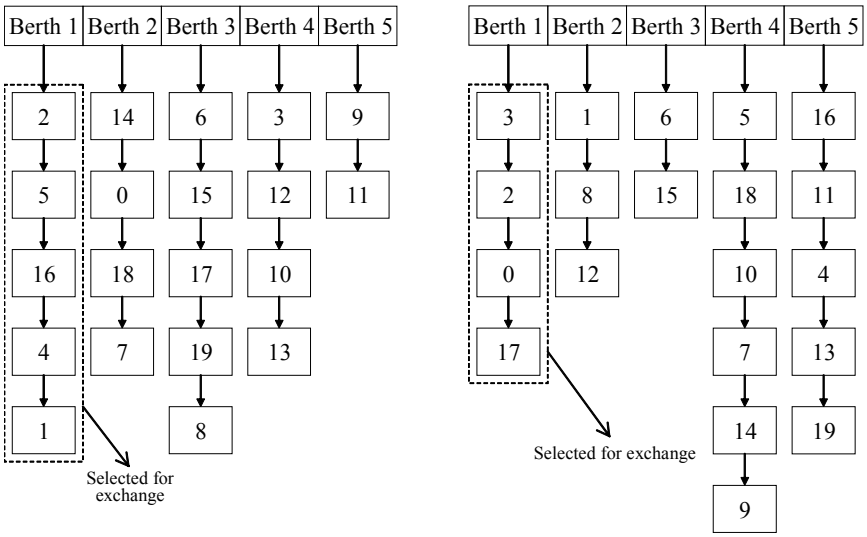
5.3.3 Population Initialization

In the population initialization process, each chromosome is formed with the predefined number of berths. Ships are then inserted into the berths such that the probability that a ship is inserted into a particular berth is inversely proportional to the handling time of the ship at the berth. The ship is inserted at a random position in the selected berth provided the insertion does not violate the physical constraints (5.6) and (5.7), otherwise another berth will be selected. This insertion process, where a ship has a higher chance of being inserted into a berth where it has a lower handling time, will be referred to as optimal berth insertion.

5.3.4 Berth-Exchange Crossover

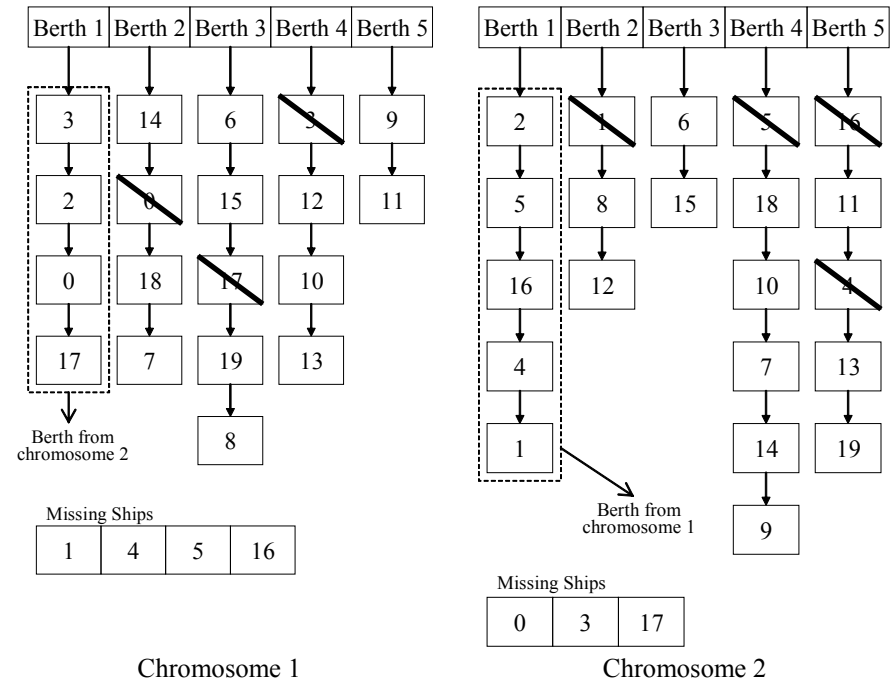
Berth-exchange crossover involves the exchange of berths between pairs of parent chromosomes, chosen based on the crossover rate, to produce offspring chromosomes. The operation of berth-exchange crossover is shown in Fig. 5.6. The berth to be exchanged is selected at random and applies to both the parent chromosomes. As the two identically indexed berths from each pair of parents have the same berth length and water depth, the exchange of the list of ships at the berth will not result in ships being allocated to a berth where they do not satisfy the physical constraints (5.6) and (5.7). Despite this, some repair work to the offspring is still required to maintain solution feasibility. Firstly, duplicated ships after crossover are removed from the offspring. These ships are removed from the original berths, while the newly acquired berth remains intact. This is followed by identifying

missing ships in each of the offspring and reinserting them back into the chromosome based on the optimal berth insertion procedure described in the previous section. The newly acquired berth is excluded from this reinsertion process unless it is the only berth that can accommodate the ship considering the physical constraints (5.6) and (5.7).

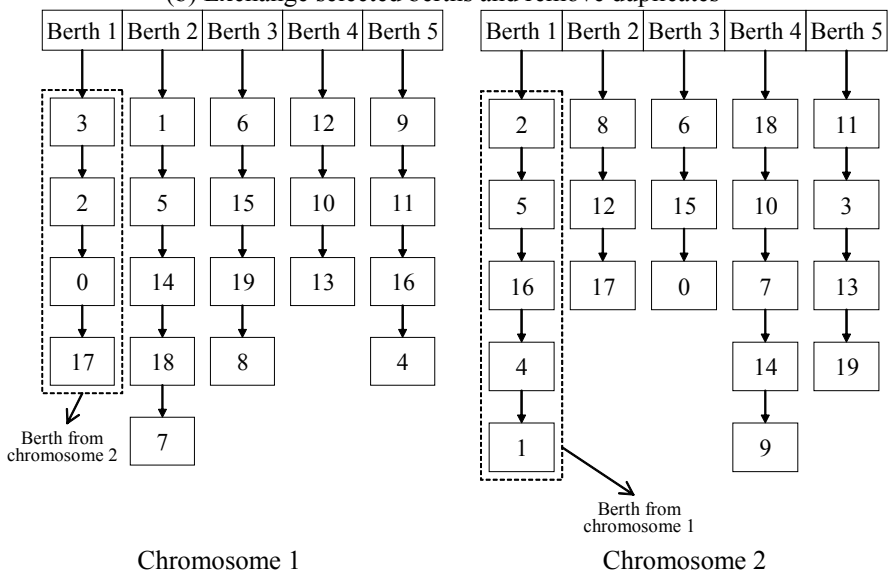


Chromosome 1 Chromosome 2

(a) Randomly select a berth for exchange



(b) Exchange selected berths and remove duplicates



(c) Reinsertion of missing ships to form offspring

Fig. 5.6 Illustration of berth-exchange crossover

One of the advantages of berth-exchange crossover is that feasibility, in terms of the physical constraints (5.6) and (5.7), is easily maintained. Duplicated ships in each offspring are easily tracked by going through the ships in the newly acquired berth

since only these ships can cause duplications. Each offspring inherits from both parents the relative order of ships within each berth since the removal of duplicated ships does not alter the order of ships. The reinsertion of missing ships provides some genetic variation.

5.3.5 Mutation

In the MOEA, chromosomes are chosen to undergo mutation with a probability equal to the mutation rate. Mutation involves removing a number of ships, randomly selected based on the reinsertion rate, from the chromosome. These ships are then reinserted back into the chromosome based on the optimal berth insertion procedure.

5.3.6 Local Search Exploitation

The MOEA utilizes a local search operator aimed at reducing the number of crossings in solutions. The operator simply involves sorting the ships assigned to a berth in accordance to their priority orders. The operator is applied to all the berths in a chromosome and the order in which the berths are being operated by the heuristic is random. The solution is stored each time a berth is sorted. At the end of the entire operation on a particular chromosome, the number of solutions stored is equal to the number of berths in the chromosome. The pool of solutions is then decoded, evaluated and ranked based on the Pareto ranking scheme. The non-dominated solutions of the pool are inserted into the original population of chromosomes before

the local search operator advances to operate on the next chromosome. After the entire original population has undergone local search, Pareto ranking is applied to the new population and the poorly ranked solutions will be removed from the population until the size of the population remains the same as before local search.

The local search operator designed is non-iterative in nature and does not compound to the computational intensity of the MOEA, which is a population-based search procedure and whose fitness evaluations are expensive due to the need to compute the values of the three considered objectives for each chromosome evaluated.

5.4 Simulation Results and Analysis

The MOEA was programmed in C++ and simulations were performed on an Intel Pentium 4 3.2 GHz computer. Table 5.1 shows the parameter settings chosen after some preliminary experiments.

Table 5.1 Parameter settings for simulation study

Parameter	Value
Population size	200
Generation number	400
Crossover rate	0.8
Mutation rate	0.3
Reinsertion rate	0.1

Since there is no commonly used benchmark for the BAP in the literature, many researchers have generated their own test problems, with a few of them using information from their ports of study. The test problems in this chapter are generated randomly but systematically. Ship arrivals are generated using an exponential distribution while ship handling times are based on a 2–Erlangian distribution. Imai *et al*. (2001) obtained these distributions from their survey on the port of Kobe. Based on the parameter settings in Table 5.2, eight test problems are generated. The characteristics of these problems are given in Table 5.3. Two extreme priority policies are experimented. First-come-first-serve (FCFS) is where ships are given priority in order of increasing arrival time, while last-come-first-serve (LCFS) is where the last ship that arrives at the port is given top priority. Although the LCFS priority order represents an impossible hypothetical port management policy, it provides a contrasting situation to the FCFS policy and is able to reveal certain characteristics of the MOEA. BAP5x100F and BAP5x100L are used for developing the algorithm, while the rest of the test problems are used to validate the performance of the proposed MOEA.

Table 5.2 Test problem parameter settings

Parameter	Characteristic
Berth length	Uniformly between 350 to 700
Berth depth	Uniformly between 40 to 60
Ship length	Uniformly between 100 and 350
Ship draft	Uniformly between 30 and 60
Ship arrival	Exponential interval with mean 12
Ship handling time	2-Erlangian distribution

Table 5.3 Characteristics of test problems

Test problem	Number of berths	Number of ships	Priority order
BAP5x100F	5	100	First-come-first-serve
BAP5x100L	5	100	Last-come-first serve
BAP5x200F	5	200	First-come-first-serve
BAP5x200L	5	200	Last-come-first serve
BAP10x100F	10	100	First-come-first-serve
BAP10x100L	10	100	Last-come-first serve
BAP10x200F	10	200	First-come-first-serve
BAP10x200L	10	200	Last-come-first serve

The subsequent sections present extensive simulation results and analysis of the proposed MOEA. Sections 5.4.1, 5.4.2, and 5.4.3, respectively, study the effects that the three primary features of local search exploitation, solution decoding scheme, and optimal berth insertion have on the quality of the generated berth schedules. The

optimization performance of the developed MOEA is then validated against a simple MOEA in Section 5.4.4.

5.4.1 Effects of Local Exploitation on Quality of Berth Schedules

The MOEA incorporates local search exploitation to complement the evolutionary operators of berth-exchange crossover and mutation, which focus on global evolutionary optimization. As described in Section 5.3.6, although the local search operator targets at reducing the number of crossings in solutions, the addition of the Pareto ranking scheme in the operator ensures that it accounts for the multi-objective nature of the problem. This section studies how the frequency of local search can affect the performance of the MOEA. At the same time, it demonstrates the effectiveness of local search in reducing the number of crossings in solutions, as well as its other implications on the quality of solutions.

Simulations were conducted by varying the frequency at which local search is performed. LS25, LS50, LS100, and LS200 are the MOEA settings where local search is applied to the evolving and archive populations every 25, 50, 100, and 200 generations, respectively. NLS is the setting which does not make use of local search at all. Ten independent runs of each of the settings were conducted on BAP5x100F to obtain statistical results.

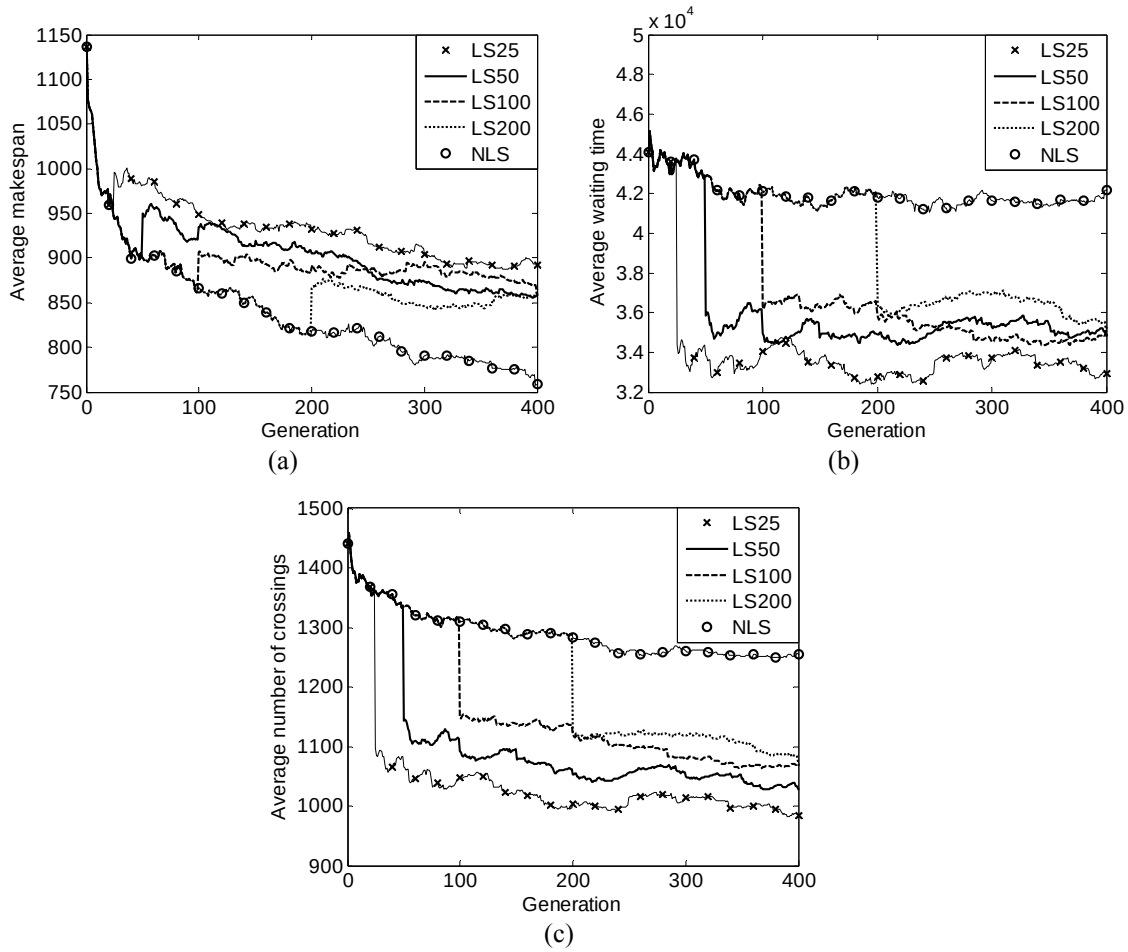


Fig. 5.7 (a) Average makespan, (b) average waiting time, and (c) average number of crossings of non-dominated solutions for different local search settings on BAP5x100F

The convergence traces of the three objectives of makespan, waiting time, and number of crossings for the five local search settings are plotted in Fig. 5.7. The convergence traces show the change in the respective objective values, averaged over all the non-dominated solutions in the archive population, over the generations. The values are further averaged over the 10 simulation runs performed. Fig. 5.7(c) shows the effectiveness of local exploitation in the MOEA in reducing the number of crossings in solutions. The local search operator causes dips in the number of

crossings whenever it is applied to solutions. Comparing the convergence traces for the different local search settings, the dips in the number of crossings get more prominent with the increase in frequency of application of the local search heuristic.

Another observation is that the dips in number of crossings coincide with the dips in waiting time in Fig. 5.7(b) and the rises in makespan in Fig. 5.7(a). As the local search operator tries to reduce the number of crossings in each solution, it indirectly reduces the waiting time but increases the makespan of the solution. This seems to suggest that the three objectives are related to one another. It appears that the objectives of makespan and waiting time are conflicting with each other, i.e. any attempt to minimize either of the objectives will cause the other objective to increase.

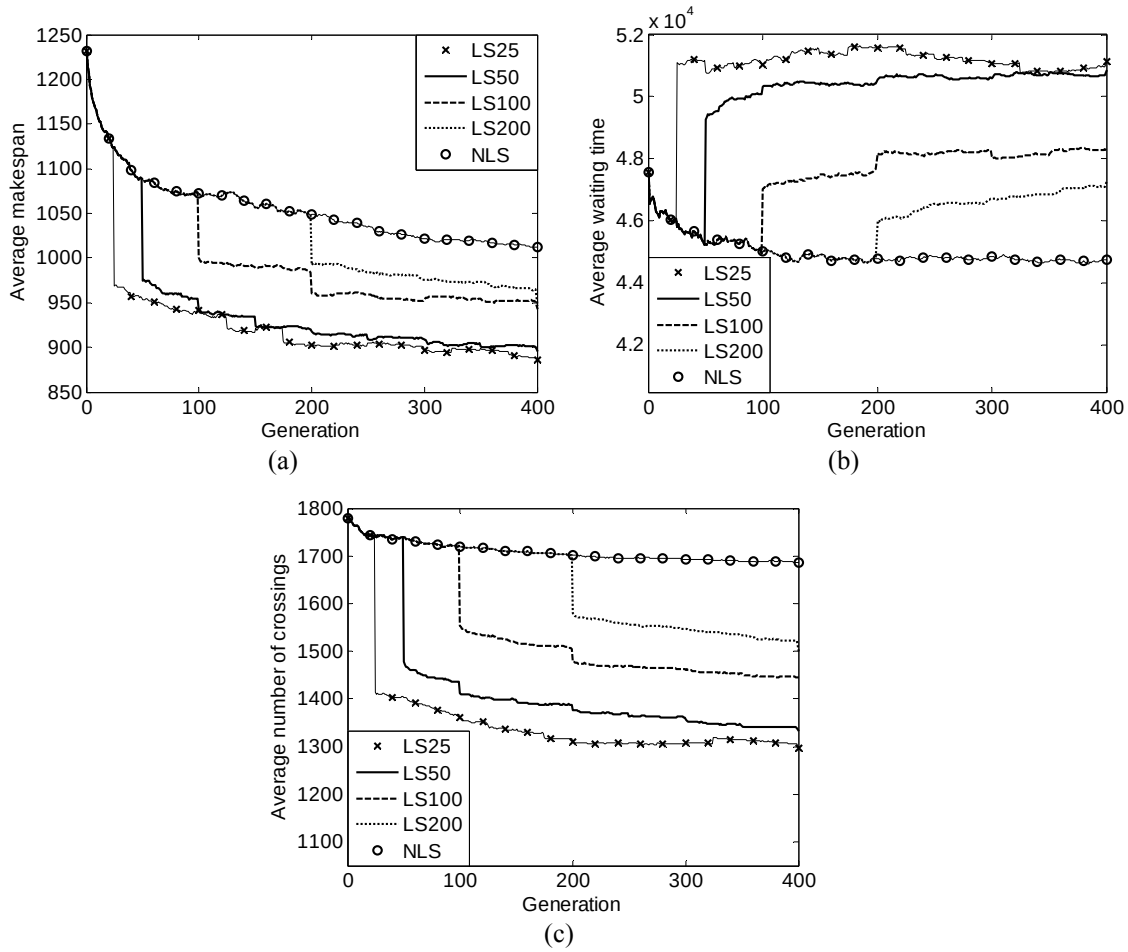


Fig. 5.8 (a) Average makespan, (b) average waiting time, and (c) average number of crossings of non-dominated solutions for different local search settings on BAP5x100L

The same simulations were also conducted on BAP5x100L and the corresponding convergence traces are plotted in Fig. 5.8. Like in Fig. 5.7(c), in Fig. 5.8(c), the local search operator causes dips in the number of crossings whenever it is applied to solutions. While this observation is expected since the operator is specifically designed to reduce the number of crossings in solutions, Fig. 5.8 does provide an interesting result. In contrast to the observation in Fig. 5.7 that a reduction in the number of crossings in a solution causes an increase in makespan and a

decrease in waiting time, Fig. 5.8 shows the exact opposite result, i.e. the reduction in the number of crossings in a solution leads to a decrease in makespan and an increase in waiting time. While the results again show that the objectives of makespan and waiting time are conflicting with each other, their relationships with the third objective have changed.

To further confirm the relationships between the three objectives in the BAP, the Pareto fronts, each of which being made up of all the non-dominated solutions in the archive population, for a random run of LS50 on BAP5x100F and BAP5x100L are plotted in Fig. 5.9(a) and Fig. 5.10(a), respectively. Separate two-dimensional graphs are also plotted for clarity in analyzing the relationships between the objectives. The plots in Fig. 5.9(c) and Fig. 5.10(c) confirm that regardless of the priority policy adopted by the port, the objectives of makespan and waiting time are conflicting with each other. On hindsight, this relation between the two objectives can be explained. In minimizing makespan, the port should delay berthing ships even when they have arrived at the port to reduce the berth idle time in between berthing of ships. In this way, ships would be waiting in the port and can berth as soon as their preceding ships have been serviced. This practice will, of course, incur the dissatisfaction of ship operators since their ships have to spend a longer time waiting at the port. The plots in Fig. 5.9(b), Fig. 5.9(d), Fig. 5.10(b), and Fig. 5.10(d) show that there is generally no fixed relation between the number of crossings in a solution and the other two objectives. However, Fig. 5.9(d) shows that the FCFS service policy leads to a proportional relationship between number of crossings and waiting time, i.e. a decrease in the number of crossings in a solution leads to a decrease in waiting time,

while Fig. 5.10(d) demonstrates that the LCFS policy leads to a conflicting relationship between the two objectives. Given the conflicting relation between makespan and waiting time, the relation between makespan and number of crossings for the LCFS policy is proportional, which can be vaguely observed in Fig. 5.10(b) since most of the solutions are situated at the bottom left and top right of the plot. This relation between makespan and number of crossings suggests that a port targeting to reduce its makespan should adopt a LCFS service policy, while the FCFS service policy benefits ship operators more in terms of lower waiting times for their ships.

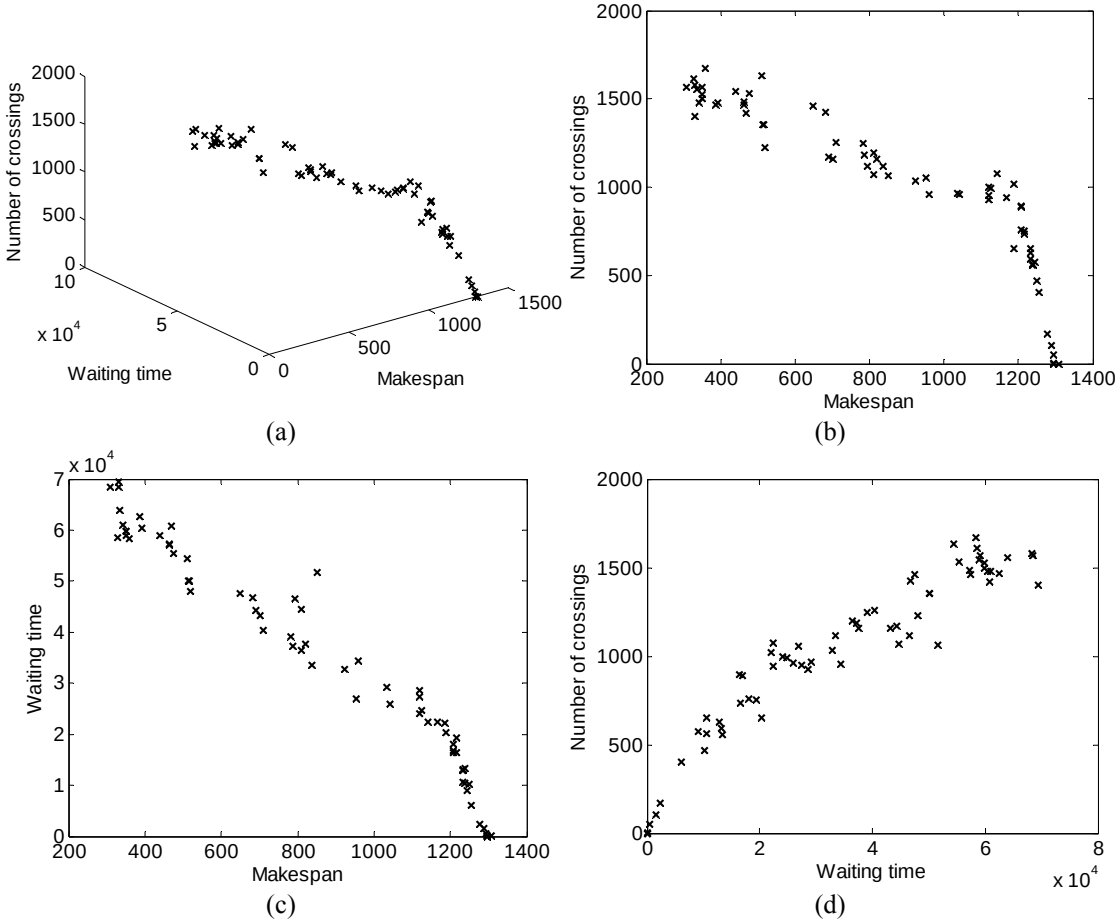


Fig. 5.9 Pareto front for a random run of LS50 on BAP5x100F

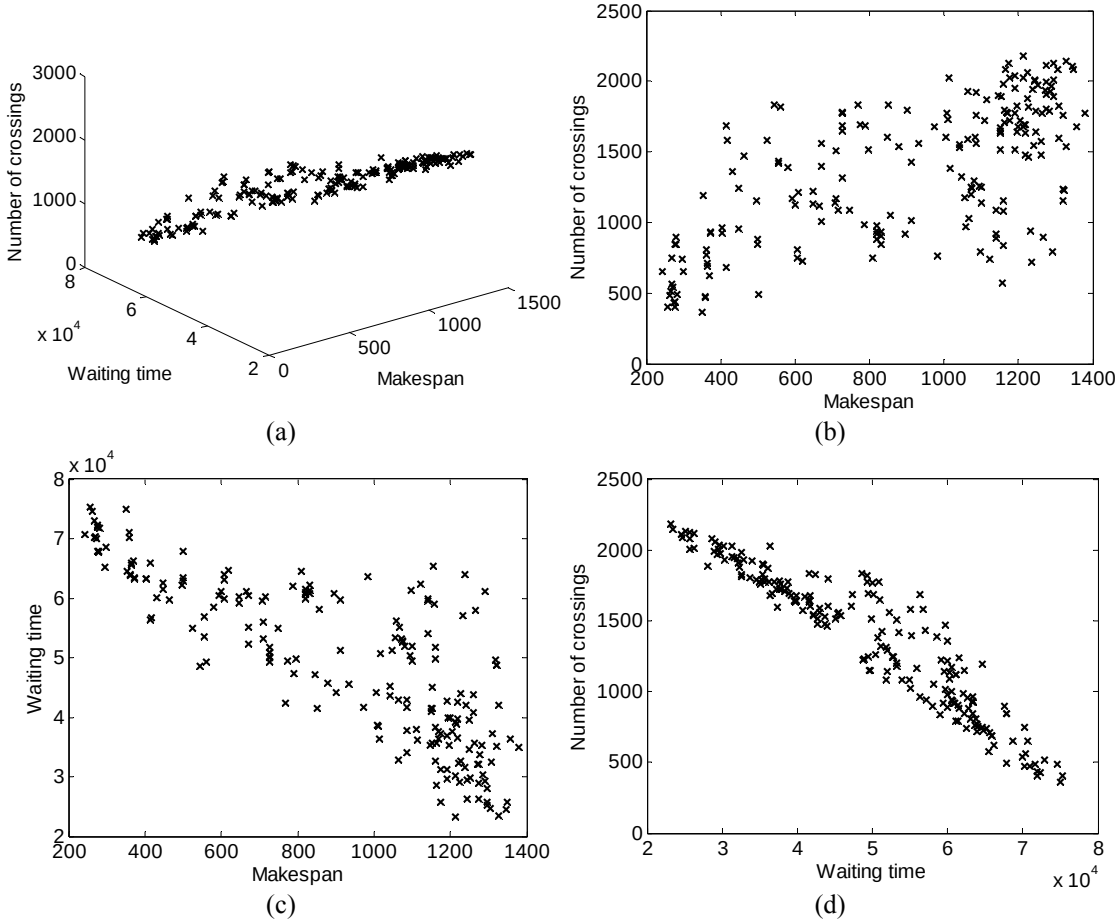


Fig. 5.10 Pareto front for a random run of LS50 on BAP5x100L

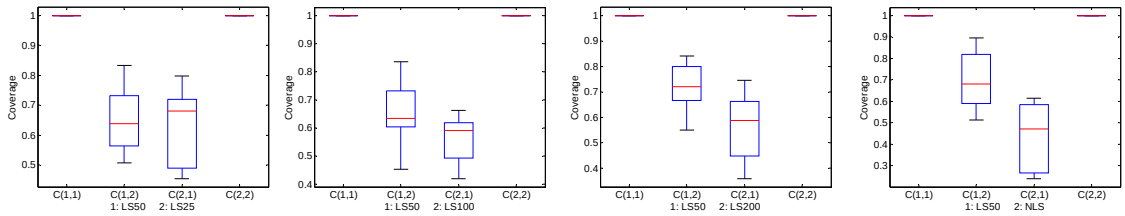
The previous results have established the relationships between the three objectives of makespan, waiting time, and number of crossings. Judging from the intricate relationships between the three objectives, it can be concluded that the BAP is inherently a multi-objective problem which needs to be solved from the perspectives of both port and ship operators. In this aspect, the MOEA, which is able to generate a Pareto set of berth schedules from which a solution that can satisfy both port and ship operators to acceptable degrees can be selected for implementation, can perform satisfactorily.

Unlike single-objective optimization, which produces a single optimal solution such that solution quality can be easily compared based on the considered objective, the solution to multi-objective optimization exists in the form of the Pareto-optimal set. As such, in comparing the performance of the different local search settings in this section, it is required to compare the optimality of their respective Pareto fronts. The optimality of Pareto fronts are usually compared based on the proximity and diversity with respect to the Pareto-optimal front (Bosman and Thierens, 2003; Deb, 2001). Proximity indicates how close a Pareto front is from the Pareto-optimal front, while diversity indicates how well-distributed and diverse the space along the Pareto front is covered with solutions. There are many multi-objective performance indicators in the literature measuring the proximity and diversity of a Pareto front. While some performance indicators require the knowledge of the Pareto-optimal front, some do not. The former are a better indication of multi-objective performance since the Pareto-optimal front provides a basis for comparison. However, it is often the case that the Pareto-optimal front is unknown and it is simply intractable to compute it, especially in large real-world combinatorial problems such as the BAP. As such, four performance indicators that do not require the knowledge of the Pareto-optimal front have been chosen in this chapter to compare multi-objective optimization performance.

The first performance indicator is the coverage function (C) (Zitzler and Thiele, 1999) which measures the proximity of a Pareto front. It is a binary quality measure which compares the dominance relationship between pairs of solution sets or Pareto fronts (Zitzler *et al*., 2003). Given a pair of solution sets (A, B), the coverage function

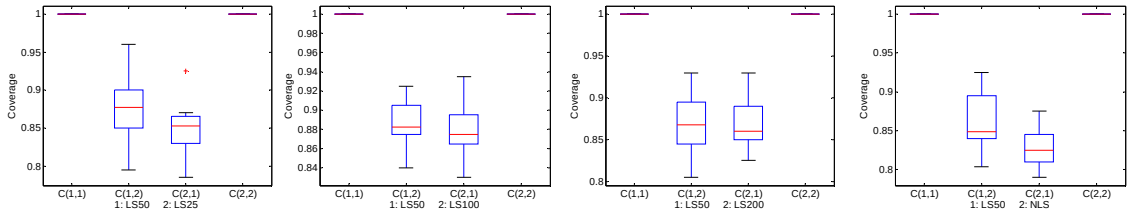
C(A, B) returns the fraction of solutions in B that are dominated by at least one solution in A. As such, if *C*(A, B) returns a value of 1, it means that all the solutions in B are dominated by or equal to the solutions in A. The other extreme case, where *C*(A, B) returns a value of 0, implies that none of the solutions in B is dominated by any of the solutions in A. It should be highlighted that both *C*(A, B) and *C*(B, A) have to be considered for a complete performance assessment. If *C*(A, B) returns a high value and *C*(B, A) returns a low value, it can be implied that the Pareto front made up of the solutions in A is closer to the Pareto-optimal front than that made up of the solutions in B.

In comparing the performance of the local search settings, due to the binary nature of the coverage function, LS50 is chosen as the basis for comparison. Since 10 independent runs of each of the settings were conducted, the comparisons are based on corresponding runs of each pair of settings, i.e. the Pareto front obtained by run number 1 of a setting is compared only with the Pareto front obtained by run number 1 of the other setting, as they share the same random number seed. The results of these comparisons are then represented in box plots and are shown in Fig. 5.11 and Fig. 5.12. Each box plot represents the distribution of the values returned by the coverage function for the 10 comparisons made for each pair of settings.



(a) (b) (c) (d)

Fig. 5.11 Coverage results for different local search settings on BAP5x100F



(a) (b) (c) (d)

Fig. 5.12 Coverage results for different local search settings on BAP5x100L

Comparing the coverage results in Fig. 5.11, on BAP5x100F, the MOEA performs better with the increase in frequency of application of the local search heuristic. In Fig. 5.11(b), Fig. 5.11(c), and Fig. 5.11(d), the difference between the medians of the coverage results gets smaller as the frequency of local search is increased with LS25 slightly surpassing the performance of LS50 in Fig. 5.11(a). In Fig. 5.12, LS50 generally performs better than the other local search settings on BAP5x100L.

Three performance indicators are used to measure the diversity of a Pareto front. The first is an adaptation of the popular maximum spread measure (Zitzler *et al.*, 2000) which indicates the maximum range of the Pareto-optimal front that is being covered by the generated solutions. Since the measure assumes the knowledge of the Pareto-optimal front, an alternative measure, which computes the volume in the

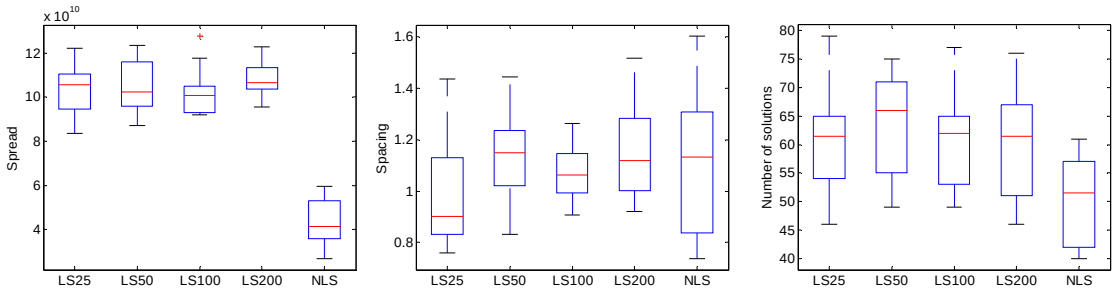
objective domain covered by the generated solutions, is used. The measure, referred to as spread, is defined in (5.15). A larger spread value implies that the solutions in the Pareto front cover a wider range of values of each of the objectives, indicating a more diverse solution set.

$$\begin{aligned}\text{Spread} = {} & \left(\text{Makespan}_{\max} - \text{Makespan}_{\min}\right)\bullet \\ & \left(\text{Waiting time}_{\max} - \text{Waiting time}_{\min}\right)\bullet \\ & \left(\text{Number of crossings}_{\max} - \text{Number of crossings}_{\min}\right)\end{aligned} \tag{5.15}$$

The next performance indicator is spacing which measures the variance of the distance of each of the solutions in the Pareto front from its nearest neighbor. Distance is measured with respect to the Euclidean distance in the three-dimensional objective space. A low spacing value implies that the solutions are more evenly distributed over the entire Pareto front. The last indicator for measuring the diversity of a Pareto front is simply the number of solutions that form the Pareto front. It gives an idea of how effective the algorithm is in generating desired solutions. In order to conclude that a Pareto front is diverse, it has to score well in all the three performance indicators. If an algorithm performs well in terms of spread and number of solutions in the Pareto front but does badly in spacing, it only means that there are many huge gaps in the Pareto front which the algorithm has failed to explore.

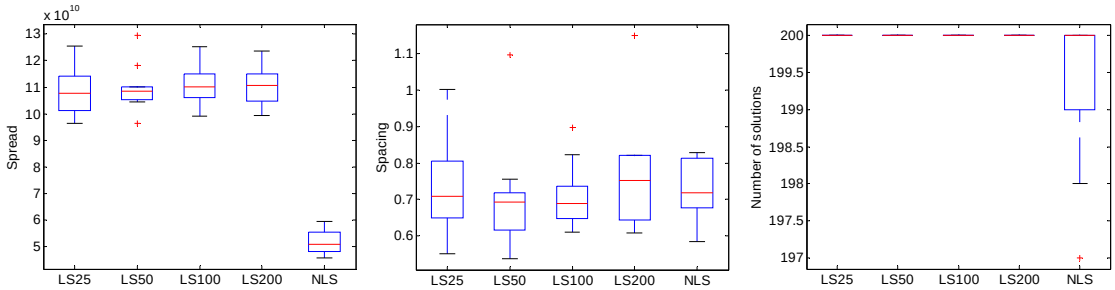
The performance of the five local search settings for the three diversity performance indicators are again represented in box plots in Fig. 5.13 and Fig. 5.14,

which show the distributions of the respective indicator values over the 10 independent simulation runs.



(a) (b) (c)

Fig. 5.13 (a) Spread, (b) spacing, and (c) number of Pareto solutions for different local search settings on BAP5x100F



(a) (b) (c)

Fig. 5.14 (a) Spread, (b) spacing, and (c) number of Pareto solutions for different local search settings on BAP5x100L

From the results in Fig. 5.13 and Fig. 5.14, it is clear that local search is beneficial to the MOEA. NLS gives poor diversity performance as the generated Pareto front has a significantly lower spread, larger spacing, and lower number of solutions compared to those generated by the other settings which make use of local search. LS50 generally performed well for the three diversity performance indicators. Coupled with its favorable proximity performance as seen in Fig. 5.11 and Fig. 5.12,

LS50 is selected as the default local search setting for any further analysis of the MOEA.

5.4.2 Effects of Solution Decoding Schemes on Quality of Berth Schedules

Two solution decoding schemes have been introduced in Section 5.3.2 to decode chromosomes into berth schedules for fitness evaluation. It has been highlighted that a chromosome may have two different sets of objective values depending on the decoding scheme applied. This section proposes a hybrid solution decoding scheme which makes use of both decoding schemes, as well as studies the effects that the two decoding schemes have on the quality of berth schedules.

Simulations were conducted using five different MOEA settings. BOD is the setting which uses solely the berthing order decoding scheme for decoding solutions, while AOD is the setting that uses only the assignment order decoding scheme. A hybrid solution decoding scheme, where each solution has a certain chance to be decoded by either of the decoding schemes, is also tested. Hybrid25, Hybrid50, and Hybrid75, respectively, are the settings where each solution has a 25%, 50%, and 75% chance of being operated by the assignment order decoding scheme, otherwise it will be operated by the berthing order decoding scheme. Like in the previous section, 10 simulation runs of each of the five settings were performed on BAP5x100F and BAP5x100L.

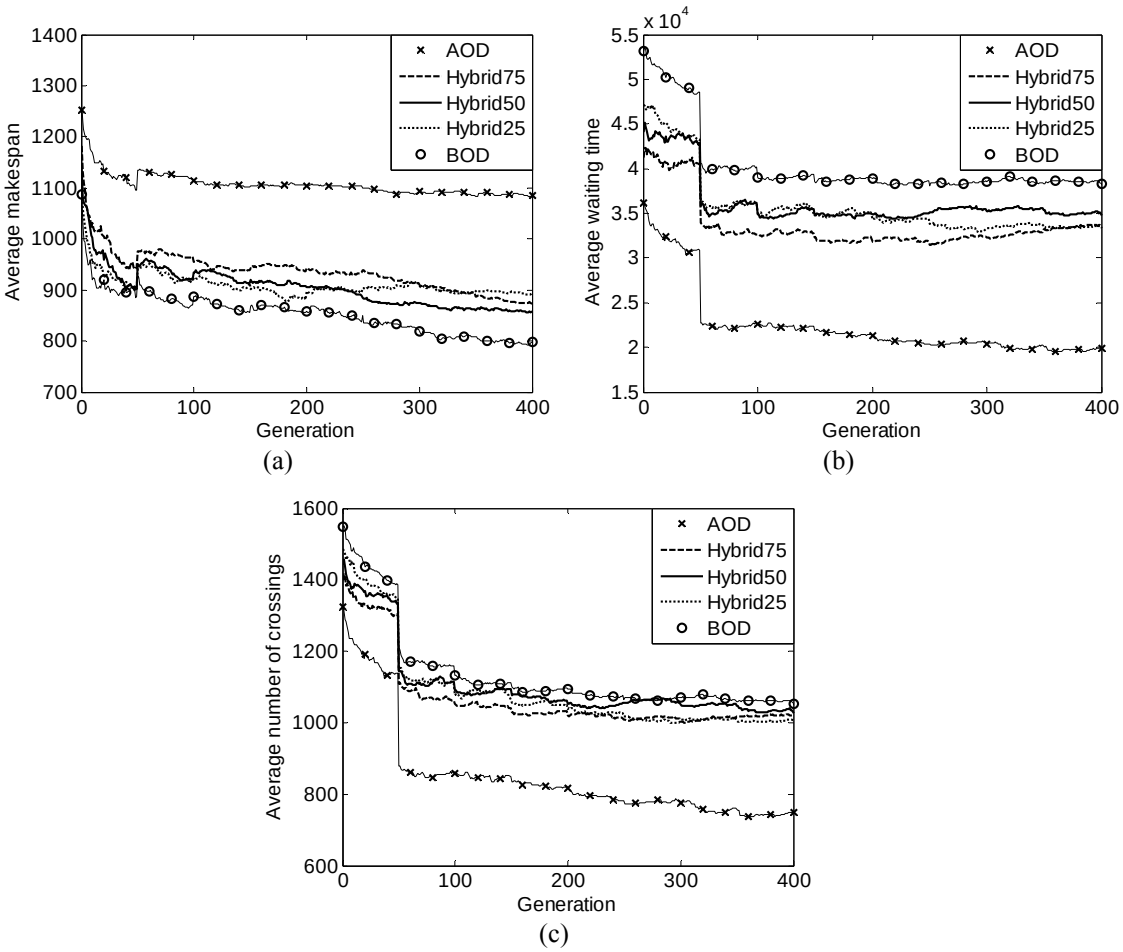


Fig. 5.15 (a) Average makespan, (b) average waiting time, and (c) average number of crossings of non-dominated solutions for different solution decoding settings on BAP5x100F

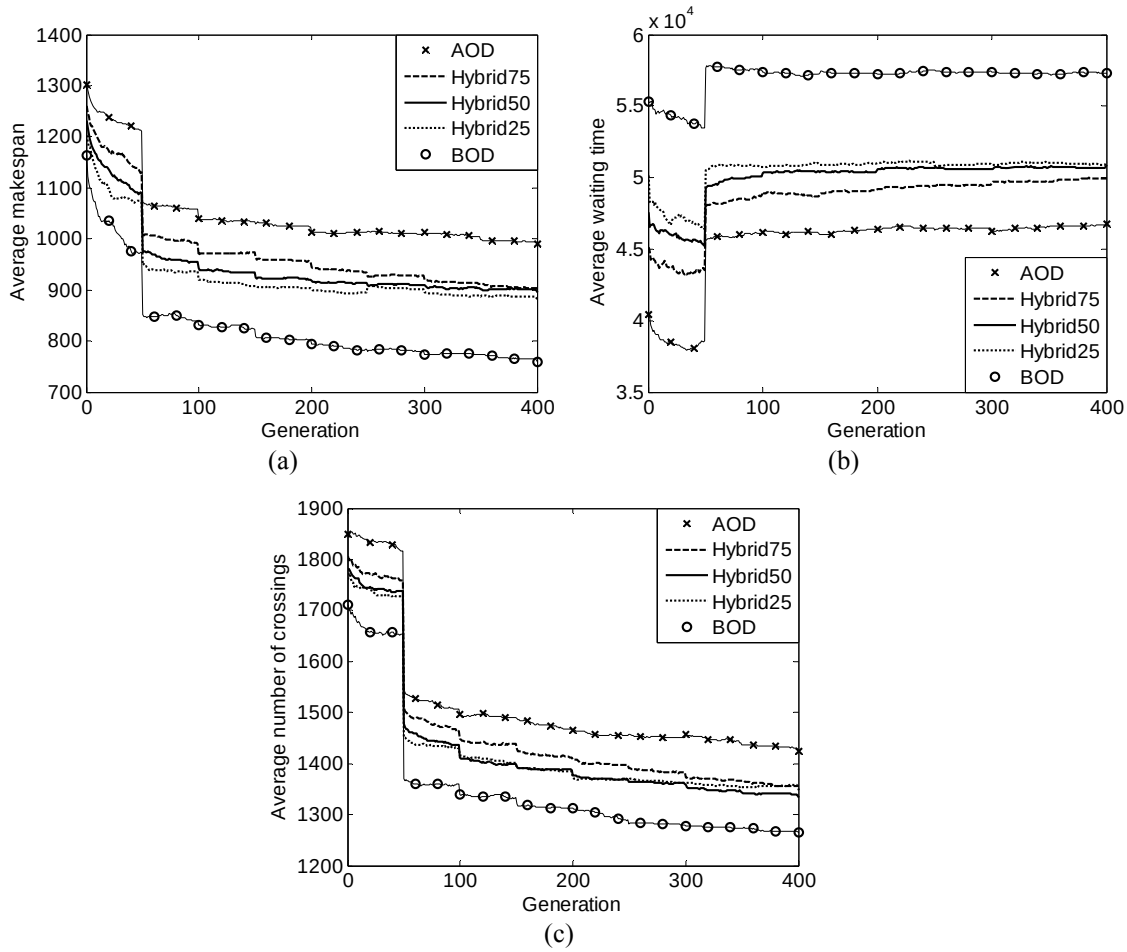


Fig. 5.16 (a) Average makespan, (b) average waiting time, and (c) average number of crossings of non-dominated solutions for different solution decoding settings on BAP5x100L

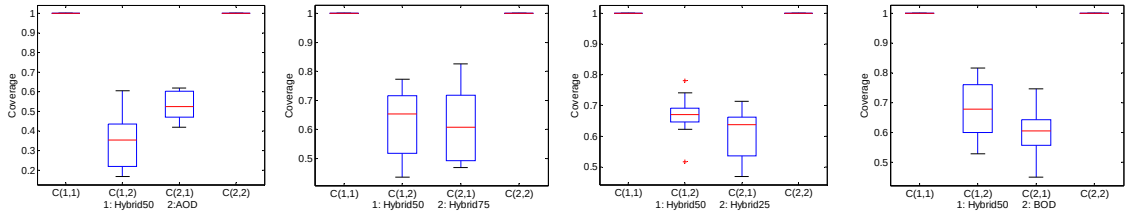
The convergence traces of the three objectives, averaged over the non-dominated solutions and over the 10 simulation runs, for the five settings on BAP5x100F and BAP5x100L are plotted in Fig. 5.15 and Fig. 5.16, respectively. It can be seen in Fig. 5.15 that AOD and BOD provide two extreme results. While AOD has a tendency of generating solutions with high makespans and low waiting times, BOD tends to concentrate on solutions with low makespans and high waiting times. The same results are also observed in Fig. 5.16 for the simulations on BAP5x100L. In terms of

number of crossings, AOD achieves better results than BOD in Fig. 5.15(c) but performs worse in Fig. 5.16(c). Given the mixed effects that the two settings have on the number of crossings in solutions, it can be inferred that the type of solution decoding scheme does not have any direct effect on the objective. Rather, the type of decoding scheme has a direct impact on the other two objectives of makespan and waiting time with the assignment order decoding scheme churning berth schedules with high makespans and low waiting times and the berthing order decoding scheme decoding solutions into schedules with low makespans and high waiting times. Any observable effect on the number of crossings in solutions is due to the underlying relationships between the three objectives for FCFS and LCFS problems that have been identified in the previous section. The effect would not be obvious if a different priority policy were adopted. One probable explanation for the berthing order decoding scheme generating berth schedules with lower makespans and higher waiting times can be made with reference to Fig. 5.4. In Fig. 5.4(a), the berth schedule generated by the berthing order decoding scheme has a lower makespan compared to that generated by the assignment order decoding scheme in Fig. 5.4(b). The berthing order decoding scheme states that succeeding ships cannot be berthed earlier than ship 4 even though they arrive at the port earlier than ship 4. This results in the berthing times of ships 1, 2, 3, and 5 to be pushed back, leading to a lower makespan. However, these ships would incur longer waiting times compared to their counterparts in Fig. 5.4(b).

Having seen the contrasting effects that the two solution decoding schemes have on the quality of berth schedules, it makes sense to use a hybrid decoding scheme that

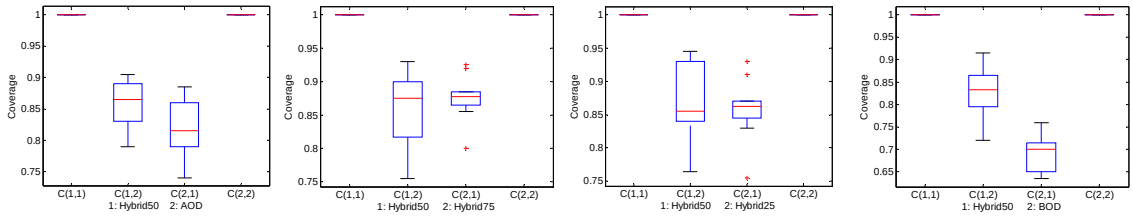
makes use of both decoding schemes. From Fig. 5.15 and Fig. 5.16, it can be seen that the three settings which make use of the hybrid decoding scheme provide intermediate results within the limits set by the extreme settings of AOD and BOD.

In order to compare the performance of the five settings, the four performance indicators introduced in the previous section are used. Hybrid50 is used as the basis of comparison for computing the coverage results. The performance comparison results are shown in Fig. 5.17, Fig. 5.18, Fig. 5.19, and Fig. 5.20.



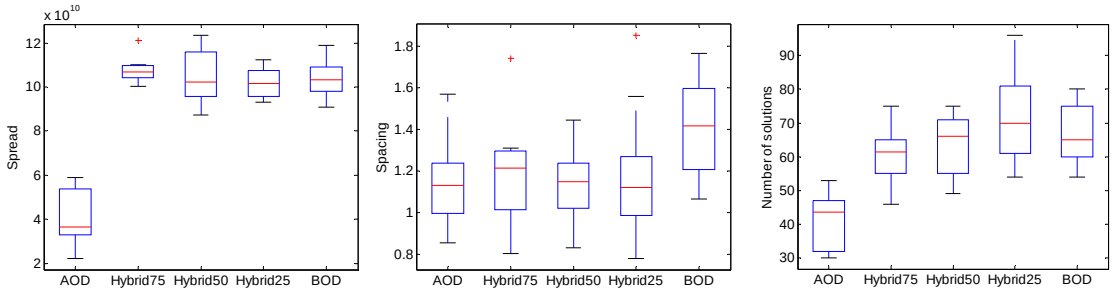
(a) (b) (c) (d)

Fig. 5.17 Coverage results for different decoding scheme settings on BAP5x100F



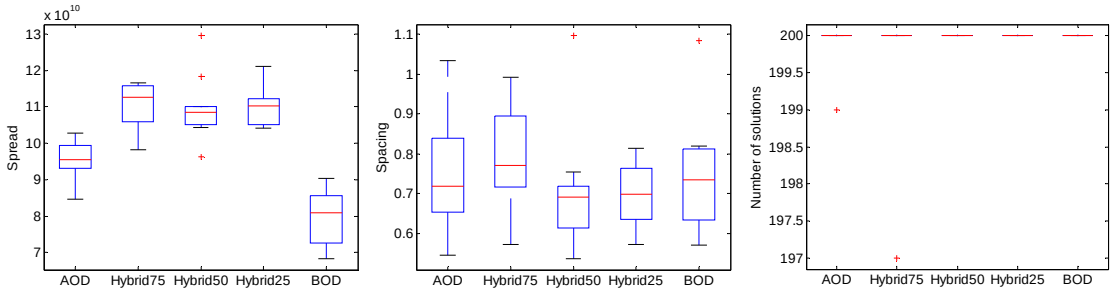
(a) (b) (c) (d)

Fig. 5.18 Coverage results for different decoding scheme settings on BAP5x100L



(a) (b) (c)

Fig. 5.19 (a) Spread, (b) spacing, and (c) number of Pareto solutions for different decoding scheme settings on BAP5x100F



(a) (b) (c)

Fig. 5.20 (a) Spread, (b) spacing, and (c) number of Pareto solutions for different decoding scheme settings on BAP5x100L

In general, the hybrid settings show better proximity and diversity results compared to AOD and BOD. There exists an abnormality though in Fig. 5.17(a), where AOD obtained a better coverage result over Hybrid50. In order to investigate the abnormality, the Pareto fronts for a random run of the two settings on BAP5x100F are plotted in Fig. 5.21.

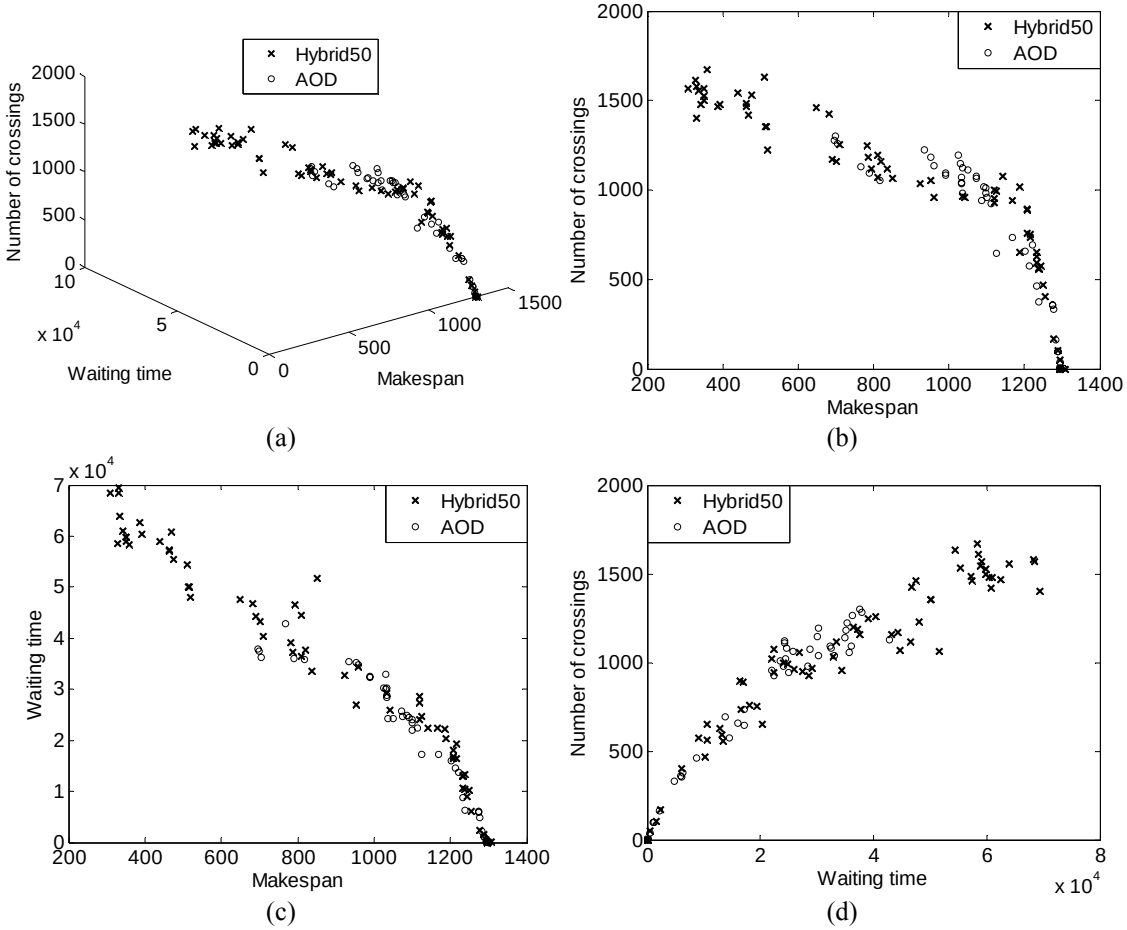
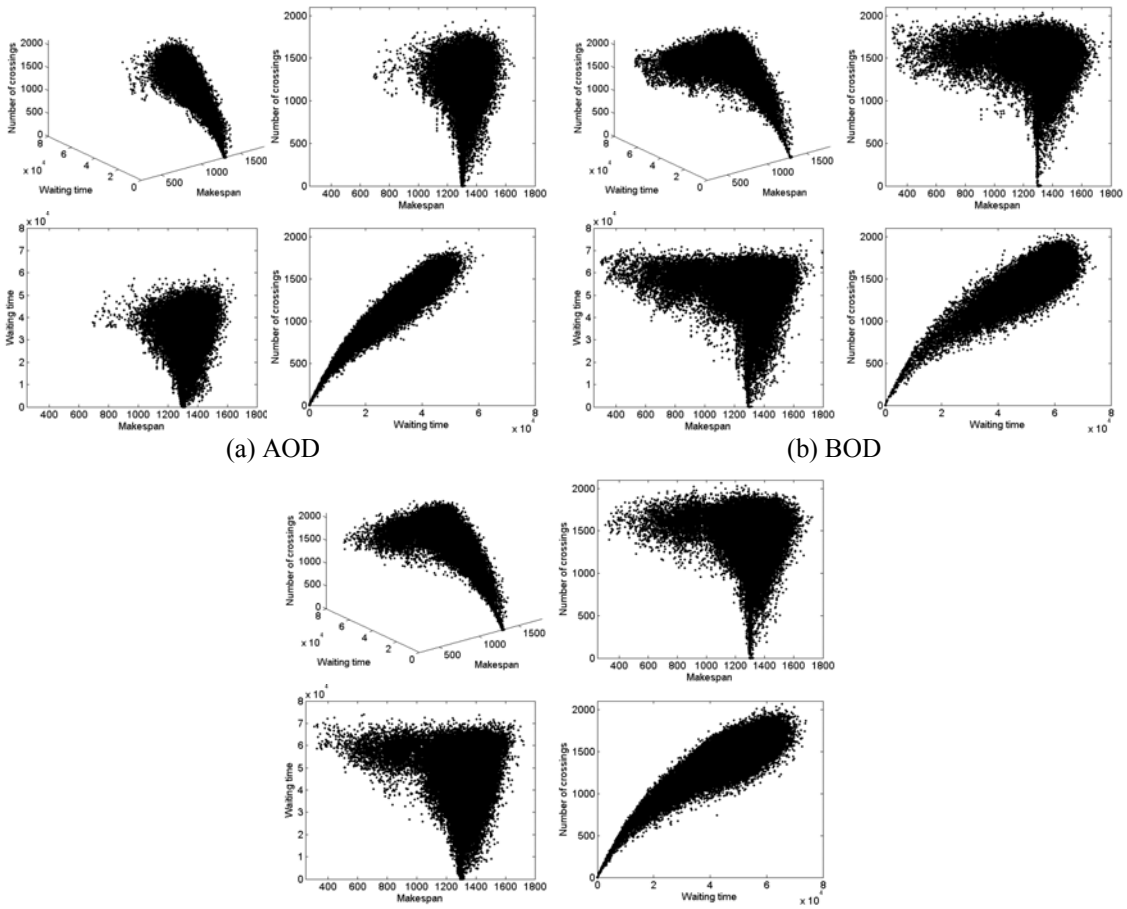


Fig. 5.21 Pareto fronts for a random run of Hybrid50 and AOD on BAP5x100F

A comparison of the Pareto fronts of Hybrid50 and AOD in Fig. 5.21 reveals a glaring deficiency in AOD. The setting is unable to locate any solution with a makespan lower than 600. It is obvious that the set of solutions generated by AOD is not as complete as that generated by Hybrid50. This explains the setting's poor performance in terms of the diversity performance indicators of spread and number of generated Pareto solutions in Fig. 5.19(a) and Fig. 5.19(c), respectively. A likely reason explaining AOD's superior proximity performance over Hybrid50 can be observed in Fig. 5.21(c). Most of the solutions generated by AOD are able to

dominate and at the same time, not being dominated by solutions generated by Hybrid50 in terms of the two objectives of makespan and waiting time. It is likely that AOD has been spending its search efforts on other areas of the search space instead of locating low makespan solutions. In order to confirm this hypothesis, the search spaces in the objective domain explored by Hybrid50, AOD, and BOD are plotted in Fig. 5.22. Each point in the plots is a point in the objective domain that has been found by the respective settings during the simulation run. For clarity in analyzing the sizes of the search spaces explored by the three settings, separate two-dimensional graphs are also plotted and the range of each of the axes is kept consistent throughout the plots.



(a) AOD (b) BOD

(c) Hybrid50

Fig. 5.22 Comparison of search spaces for different decoding scheme settings on BAP5x100F

Comparing the search spaces of AOD and BOD in Fig. 5.22(a) and Fig. 5.22(b), it can be observed that certain parts of the search space that AOD has explored have been left out by BOD and vice versa. To allow a better visual comparison, the corresponding two-dimensional search space plots are superimposed onto each other in Fig. 5.23. Judging from the search spaces that both settings have left unexplored, it is quite clear that the search space explored by Hybrid50 in Fig. 5.22(c) is a union of the search spaces covered by AOD and BOD. The hybrid setting is able to benefit from the complementary behavior of the two solution decoding schemes, which

allows a larger search space to be explored. This advantage translates into better proximity and diversity results as have been observed in Fig. 5.17, Fig. 5.18, Fig. 5.19, and Fig. 5.20. Since the proximity and diversity results for the three hybrid settings are relatively comparable, Hybrid50 is chosen as the default setting for any subsequent analysis of the MOEA.

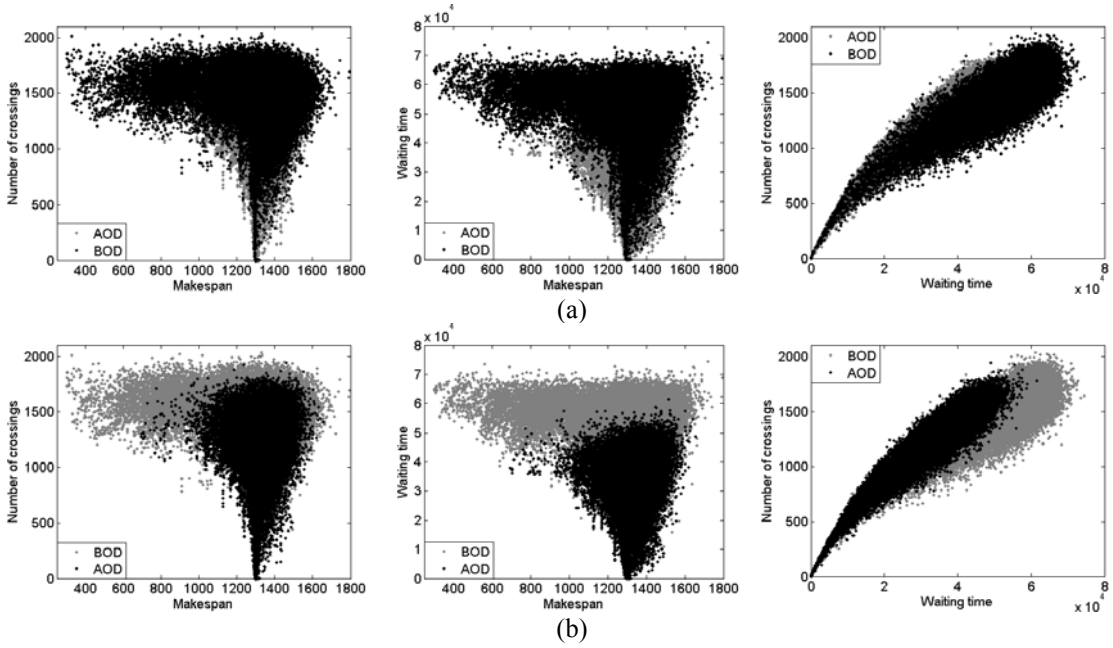


Fig. 5.23 Superimposing search space plots of (a) BOD onto AOD and (b) AOD onto BOD

5.4.3 Effects of Optimal Berth Insertion on Quality of Berth Schedules

Optimal berth insertion is utilized during population initialization, berth-exchange crossover, and mutation to insert ships into chromosomes. The insertion procedure gives each ship a higher chance of being inserted into a berth where it has a lower handling time. This section presents the performance of the MOEA with and without

the insertion procedure. In the case where optimal berth insertion is not used, each ship has equal chance of being inserted into any of the berths. This setting will be known as RAND. Ten independent simulation runs of the MOEA and RAND were performed on BAP5x100F and BAP5x100L. Since the optimal berth insertion procedure targets to minimize the handling times of ships, the convergence traces of the total handling time incurred by the entire fleet of ships, averaged over the non-dominated solutions and over the 10 simulation runs, for the two settings on BAP5x100F and BAP5x100L are plotted in Fig. 5.24.

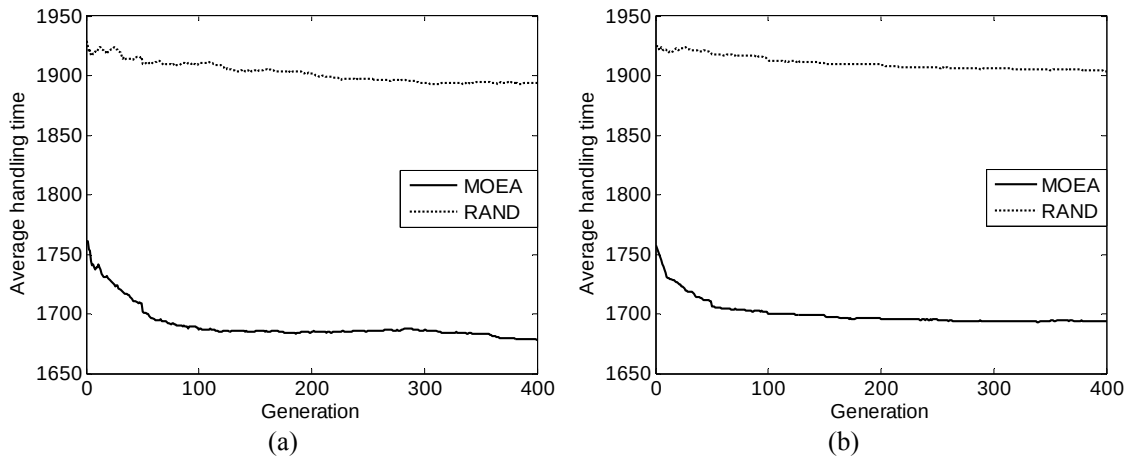
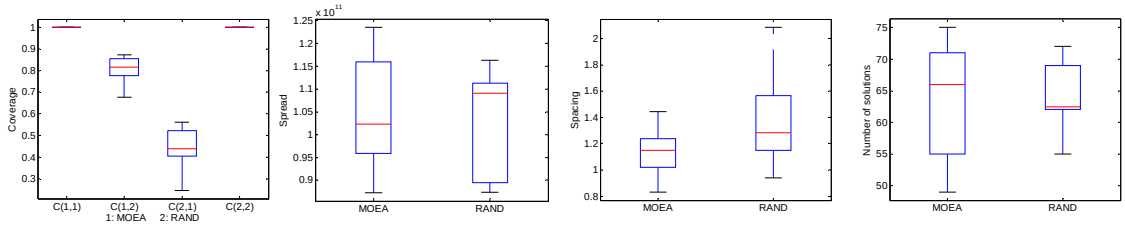


Fig. 5.24 Average handling time of non-dominated solutions for MOEA and RAND on (a) BAP5x100F and (b) BAP5x100L

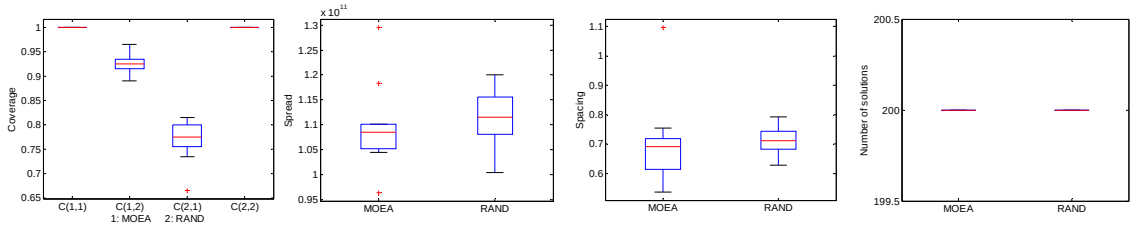
It is obvious from Fig. 5.24 that the optimal berth insertion procedure has achieved its aim of reducing the handling times of ships. In most situations, the reduction in handling time translates directly to a reduction in makespan and waiting time since handling time is a component of the two objectives. The lower average handling time for the MOEA at generation 0 shows the positive effect that optimal

berth insertion has in population initialization. The steeper decline in average handling time for the MOEA in the initial stages of evolution is due to the incorporation of optimal berth insertion in berth-exchange crossover and mutation. To compare the multi-objective optimization performance of the two settings, the four proximity and diversity performance indicators are computed for the Pareto fronts generated by the two settings. The comparison results are plotted in Fig. 5.25 and Fig. 5.26.



(a) (b) (c) (d)

Fig. 5.25 (a) Coverage, (b) spread, (c) spacing, and (d) number of Pareto solutions for MOEA and RAND on BAP5x100F



(a) (b) (c) (d)

Fig. 5.26 (a) Coverage, (b) spread, (c) spacing, and (d) number of Pareto solutions for MOEA and RAND on BAP5x100L

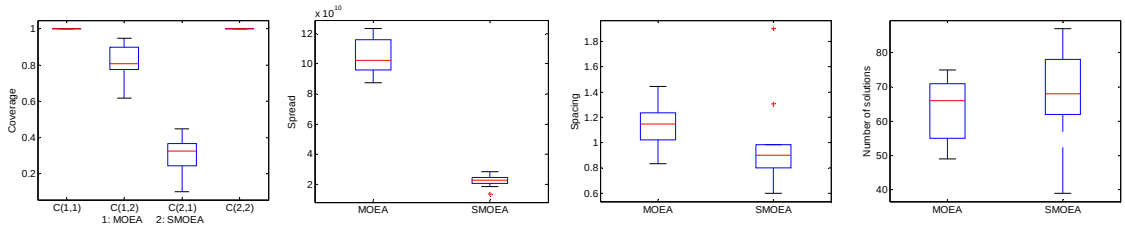
From Fig. 5.25(a) and Fig. 5.26(a), it can be seen that the MOEA is superior to RAND in terms of the proximity performance measure. However, the three performance metrics of spread, spacing, and number of generated Pareto solutions

indicate that the two settings have comparable diversity performance. This finding implies that the optimal berth insertion procedure focuses more on improving the proximity of the generated Pareto front. Unlike local search and the hybrid solution decoding scheme, it has little effect on the diversity of the obtained Pareto front.

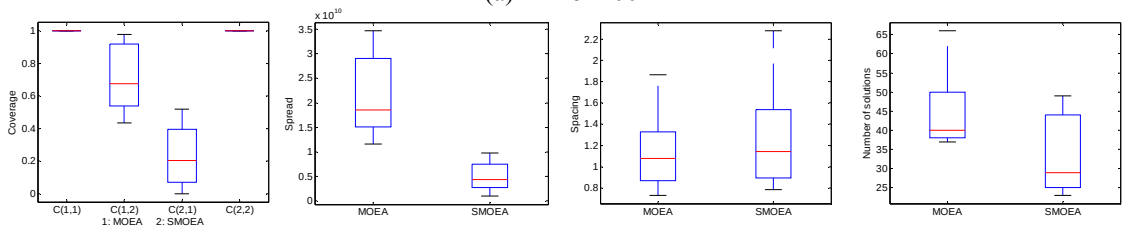
5.4.4 Performance of MOEA on other Test Problems

The previous sections have studied how the three primary features of the MOEA affect the quality of berth schedules. While local search reduces the number of crossings in solutions, the hybrid solution decoding scheme is able to exploit on the advantages of the berthing and assignment order decoding schemes to allow a larger search space to be explored, leading to a Pareto front that is superior in terms of proximity and diversity. Lastly, optimal berth insertion reduces the handling times of ships which in turn reduces their waiting times and the makespan of the port, further improving the proximity of the Pareto front. This section validates the optimization performance of the proposed MOEA against a simple MOEA (SMOEA) on the test problems listed in Table 5.3. SMOEA has the same functions as the MOEA except that it does not make use of the three proposed features. To decode solutions for fitness evaluation, SMOEA uses the berthing order decoding scheme.

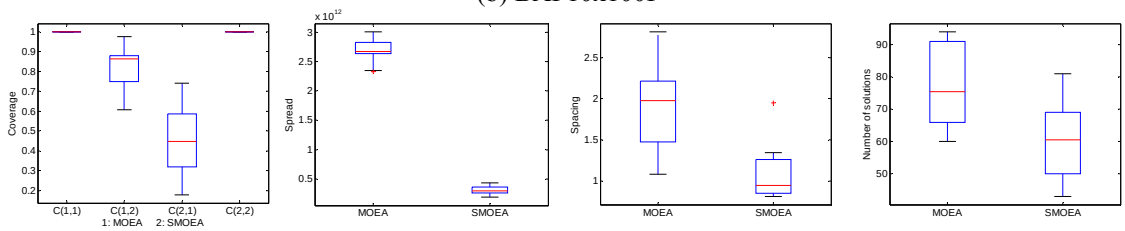
The performance comparison results of the MOEA and SMOEA are shown in Fig. 5.27 and Fig. 5.28.



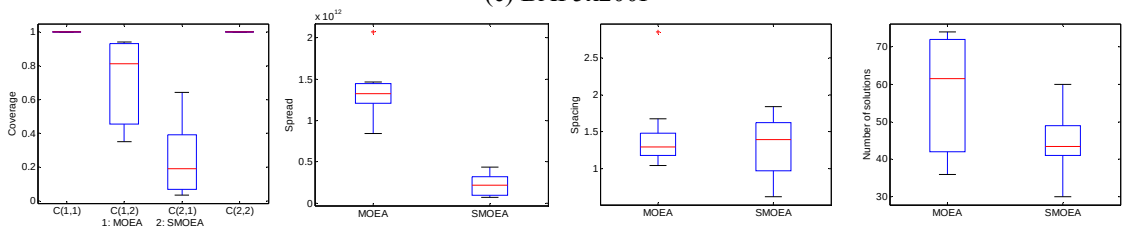
(a) BAP5x100F



(b) BAP10x100F



(c) BAP5x200F



(d) BAP10x200F

Fig. 5.27 Performance comparison between MOEA and SMOEA on FCFS test problems

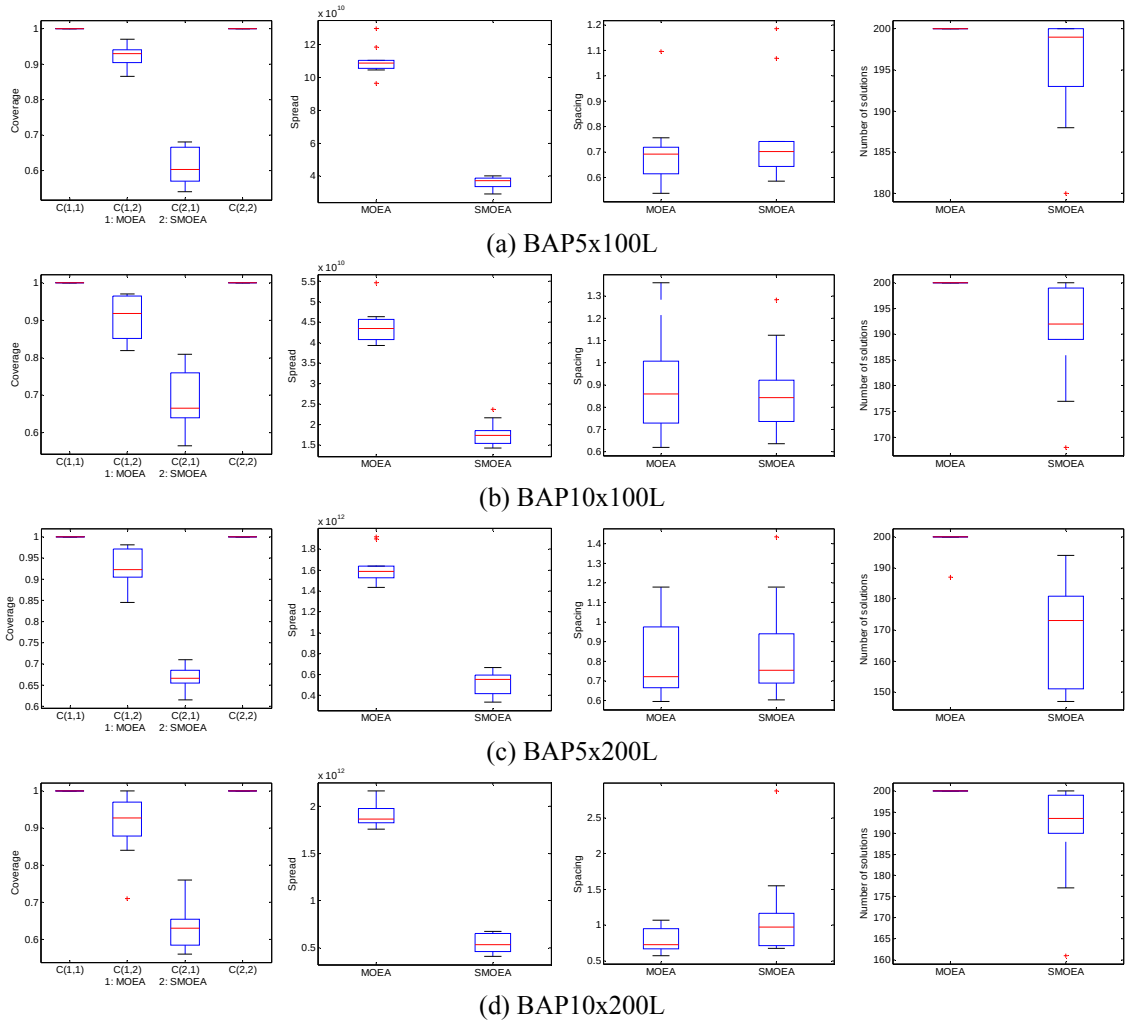


Fig. 5.28 Performance comparison between MOEA and SMOEA on LCFS test problems

The comparison results show that the MOEA consistently outperforms SMOEA in terms of coverage and spread. While the MOEA is generally comparable to SMOEA in terms of spacing and superior in terms of number of Pareto solutions generated, there are a few minor exceptions. For BAP5x100F, some of the simulation runs of SMOEA are able to generate more solutions than those of the MOEA. SMOEA also performs better in terms of spacing for that test problem. A closer examination of the Pareto fronts for a random run of the MOEA and SMOEA on

BAP5x100F in Fig. 5.29 reveals the superiority of the MOEA despite the slightly negative results in spacing and number of Pareto solutions generated. Although the Pareto front obtained by SMOEA consists of more solutions (78 solutions for SMOEA compared to 67 for the MOEA), most of the solutions are inferior and dominated by the solutions in the Pareto front generated by the MOEA. From the comparison results in Fig. 5.27 and Fig. 5.28, it is evident that the three features of local search, hybrid solution decoding scheme, and optimal berth insertion play an important role in the optimization performance of the proposed MOEA.

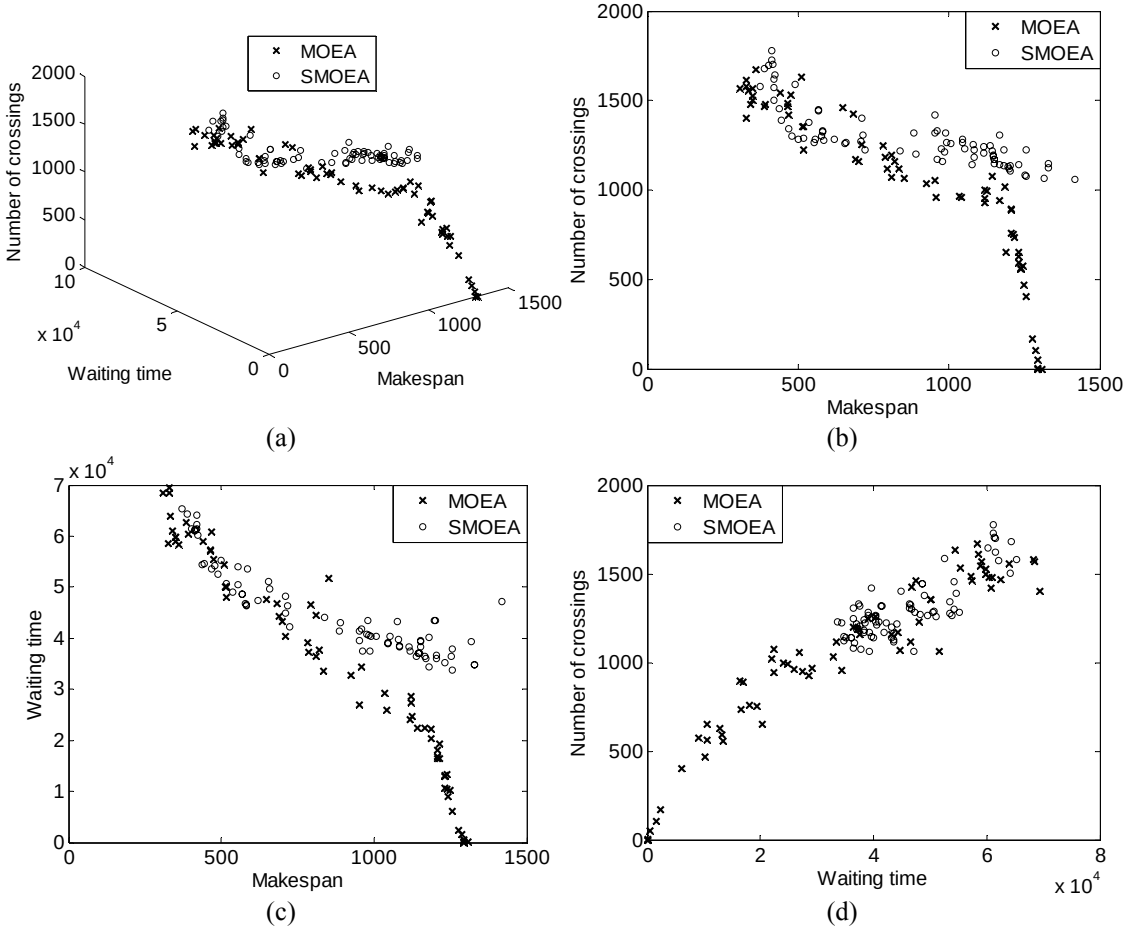


Fig. 5.29 Pareto fronts for a random run of the MOEA and SMOEA on BAP5x100F

5.5 Summary

A berth allocation problem (BAP), which requires the determination of exact berthing times and positions of incoming ships in a container port, has been studied in this chapter. The problem involves the optimization of berth schedules so as to minimize concurrently the three objectives of makespan, waiting time, and degree of deviation from a predetermined priority schedule. These objectives represent the interests of both port and ship operators. Unlike most existing approaches in the literature, which

are single-objective-based, a multi-objective evolutionary algorithm (MOEA), which incorporates the concept of Pareto optimality, has been proposed in this chapter to solve the multi-objective BAP. The MOEA is equipped with three primary features which have been specifically designed to target the optimization of the three objectives. The features include a local search heuristic, a hybrid solution decoding scheme, and an optimal berth insertion procedure. The effects that each of these features has on the quality of berth schedules have also been studied in this chapter.

Chapter 6

Multi-Objective Optimization in Vehicle Routing Problem with Stochastic Demand

A multi-objective vehicle routing problem with stochastic demand (VRPSD) is considered in this chapter. The VRPSD involves the routing of a set of identical vehicles with limited capacity from a central depot to a set of geographically dispersed customers to satisfy their demands. Unlike the exam timetabling problem (ETTP) and the berth allocation problem (BAP) studied in the previous chapters, where all aspects of the problem are known at the point of solving the problem, the VRPSD is a stochastic optimization problem and some problem parameters are uncertain during the solution-searching process. In the VRPSD, the actual demand of each customer is unknown during the routing process but is revealed only when the vehicle actually reaches the customer. The absence of information about the actual customer demands poses a problem to the fitness evaluation component of the multi-

objective evolutionary algorithm (MOEA) since it is not possible to tell exactly when a vehicle needs to be restocked. The fitness evaluation component of the MOEA needs to be modified to adapt the MOEA to solve the stochastic optimization problem.

6.1 Introduction

The vehicle routing problem (VRP) is a generic name referring to a class of combinatorial optimization problems in which customers are to be served by a number of vehicles. The vehicles leave the depot, serve customers in the network and on completion of their routes, return to the depot. Each customer is described by a certain demand. Other information includes the co-ordinates of the depot and customers, the distance between them, and the capacity of the vehicles providing the service. All these information are known in advance for the purpose of planning a set of routes which minimizes transportation cost while satisfying capacity constraints (Prins, 2004), time constraints (Hwang, 2002; Thangiah *et al*., 1996), and time window constraints (Lee *et al*., 2003; Potvin and Bengio, 1996; Tan *et al*., 2001a, 2001b, 2003b). However, in many real-world applications, one or more parameters of the VRP tend to be random or stochastic in nature, giving rise to the stochastic vehicle routing problem (SVRP).

There are three basic classes of SVRP (Gendreau *et al*., 1996a; Laporte and Louveaux, 1998; Yang *et al*., 2000): stochastic customers (Bertsimas *et al*., 1990; Jaillet, 1987; Jaillet and Odoni, 1988), stochastic demands, and stochastic travel and service times. This chapter considers the capacity and time constrained vehicle

routing problem with stochastic demand (VRPSD), where only the customer demand is stochastic and all other parameters are known *a priori*. This problem appears in the delivery of home heating oil (Dror *et al*., 1985), trash collection, sludge disposal (Larson, 1988), beer and soft drinks distribution, the provision of bank automates with cash, and the collection of cash from bank branches (Lambert *et al*., 1993).

The VRPSD differs from its deterministic counterparts in that when some data are random, it is no longer possible to require that all constraints be satisfied for all realizations of the random variables (Laporte and Louveaux, 1998). The basic characteristic of the VRPSD is that the actual demand of each customer is revealed only when the vehicle reaches the customer. As such, on one hand, the vehicle routes are designed in advance by applying a particular algorithm but on the other hand, due to the uncertainty of demands at the customers, at some point along a route the capacity of the vehicle may be depleted before all demands on the route have been satisfied. Dror and Trudeau (1986) and Teodorović and Lucić (2000) referred to such a situation as "route failure". In the capacity constrained VRPSD, recourse or corrective actions, e.g. making a return trip to the depot to restock, have to be designed to ensure feasibility of solutions in case of route failure.

In the time constrained VRPSD, one possible corrective action is to apply a penalty when the duration of a route exceeds a given bound. This penalty would correspond to the overtime pay that a driver receives. As such, the situation of route failure, together with all its associated recourse policies, would definitely generate additional transportation cost, in terms of the travel distance for the to and fro trips to the depot and the overtime pay for drivers, which are stochastic in nature. This means

that the actual cost of a particular solution to the VRPSD cannot be known with certainty before the actual implementation of the solution. One of the main obstacles to solving the VRPSD is in finding an objective function which takes into consideration all these costs. It is for this reason that Dror (1993), Gendreau *et al*. (1996a), and Laporte and Louveaux (1998) agree that the VRPSD and the SVRP in general are inherently much more difficult to solve than their deterministic counterparts.

From the studies of Yang *et al*. (2000), it is clear that the VRPSD is inherently a multi-objective optimization problem. In minimizing the expected transportation cost, in terms of travel distance, of a particular solution, an algorithm for the VRPSD must also account for the feasibility of implementation of the solution in terms of the duration of the routes, i.e. both capacity and time constraints must be considered. According to Yang *et al*. (2000), a single, long route has the lowest expected travel distance but it may not be feasible to implement in the context of the real-world. Therefore, it is required to minimize multiple conflicting cost functions, such as the travel distance, the remuneration for drivers including overtime pay, and the number of vehicles required, concurrently, which is best solved by means of multi-objective optimization. Most of the existing literature, however, either do not consider time constraints or use single-objective-based heuristic methods that incorporate penalty functions or combine the different objectives by a weighting function. Furthermore, as mentioned earlier, one of the main difficulties of solving the VRPSD is finding an objective function that is able to define properly the expected transportation cost of a solution, which includes the initial cost of travel before route failures occur as well as

the additional cost generated by recourse policies in case of route failures. These characteristics of the VRPSD must definitely be addressed when solving the problem.

Like in the previous two chapters, the multi-objective evolutionary algorithm (MOEA) is applied to solve the multi-objective VRPSD optimization problem. The algorithm is featured with two VRPSD-specific heuristics, which are based on two route structures of a solution to the VRPSD identified by Dror and Trudeau (1986), for local exploitation in the evolutionary search. In addition, an intuitive route simulation method (RSM) is proposed to address the issue of evaluating the expected costs of solutions. A procedure based on the RSM is also proposed to assess the quality of solutions on top of comparing their expected transportation costs, which has been used as the main performance measure hitherto.

This chapter is organized as follows: Section 6.2 gives an overview of existing works as well as the problem formulation of the VRPSD. Section 6.3 presents the problem-specific features that adapt the MOEA for solving the VRPSD. Section 6.4 presents the extensive simulation results and analysis of the proposed algorithm. Conclusions are drawn in Section 6.5.

6.2 Background Information

6.2.1 Overview of Existing Works

Many researchers have studied the VRPSD in two frameworks, namely as a chance constrained program (CCP) (Charnes and Cooper, 1959, 1963) or as a stochastic

program with recourse (SPR). In a CCP, the problem consists of designing a set of vehicle routes for which the probability of route failure is constrained to be below a certain threshold. It was shown by Steward and Golden (1983) that, under some restrictive assumptions, the problem can be reduced to a deterministic VRP and then solved using existing deterministic algorithms. Although the CCP tries to control the probability of route failure, the cost of such failures is ignored. In contrast, the SPR tries to minimize the expected transportation cost, which includes the travel cost as well as the additional cost generated by recourse policies. Gendreau *et al*. (1996a) commented that SPRs are typically more difficult to solve than CCPs but their objective functions are more meaningful. As such, most of the recent researches revolve around SPRs and results obtained are compared and assessed based on the expected transportation costs of solutions. In using the SPR, various recourse policies have been explored and there are three common recourse policies (Gendreau *et al*., 1996a; Laporte and Louveaux, 1998). In the first approach, also known as the simple recourse policy (Gendreau *et al*., 1995, 1996b; Laporte and Louveaux, 1993, 1998; Teodorović and Pavković, 1996; Teodorović and Lucić, 2000), a vehicle returns to the depot to restock when its capacity becomes attained or exceeded. The vehicle will then resume service at the customer on the planned route where route failure had occurred. In the second approach (Bertsimas *et al.,* 1995; Bianchi *et al*., 2004, 2006; Yang *et al*., 2000), preventive restocking is planned at strategic points, preferably when the vehicle is near to the depot and its capacity is almost empty, along the scheduled route instead of waiting for route failures to occur. The third approach

sought to optimize the remaining portion of a route after each failure or knowledge of the actual demand of each customer (Secomandi, 2000; 2001).

Yang *et al*. (2000) proposed a dynamic programming recursive objective function for the VRPSD. The Or-opt operator is adapted to the stochastic case using a fast approximation computation for the change in the objective function when performing a local search move where the objective function needs to be repeatedly computed. Yang *et al*. (2000) also showed that the optimal route, in terms of travel distance, is always a single route, if only capacity constraints are considered.

Bianchi *et al*. (2004, 2006) also employed the recursive objective function and its approximation in Or-opt operations in the analysis of various meta-heuristics such as iterated local search, tabu search, simulated annealing, ant colony optimization, and evolutionary algorithm. However, it should be noted that the dynamic programming recursive objective function (Bianchi *et al.,* 2004, 2006; Yang *et al.,* 2000) is applicable only if demands take on integer values, i.e. the stochastic demands follow discrete distributions.

Dror and Trudeau (1986) showed that given that the customers' demands are independent random variables with non-negative means, route failures are more likely to occur at the end of a route. They also showed that the expected transportation cost of a route is dependent on the direction in which the route is traversed.

6.2.2 Problem Formulation

This section presents the mathematical model of the VRPSD. The time and capacity constrained problem, as well as the recourse policy used, will also be explained. Fig. 6.1 shows a complete graph representing a model of a simple VRPSD and its solution. The solution consists of two routes, R_1 and R_2, connecting the depot to a set of customers which are each identified by a number. For a particular route, the arrows show the sequence in which the customers will be visited by the vehicle and the route must start and end at the depot.

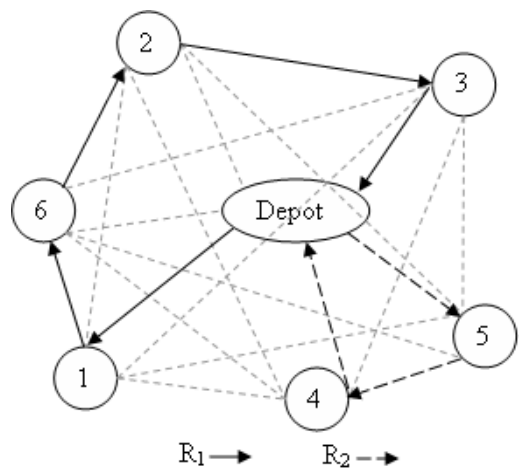


Fig. 6.1 Graphical representation of a simple vehicle routing problem

Definitions of some of the frequently used notations for the VRPSD, leading to the formulation of the mathematical model of the problem, are given as follows.

1) Customers and depot: The customer set $V = \{0,1,2,...,N_C\}$ represents the N_C customers to be visited. For simplicity, the depot is denoted as customer 0, or v_0. The depot is treated as the source of service demanded by the customers. Every

vehicle must start and end its route at the depot. With the exception of the depot, each customer v_i has a demand distribution D_i. D_i is a normal random variable and is described by two parameters, the mean μ_i and the variance σ_i^2. The actual demand of each customer d_i is revealed when the vehicle first arrives at the customer. There is also a service time s_i associated with each customer and the depot, which will be incurred each time the vehicle arrives at the customer or returns to the depot for restocking.

2) Node: A node is denoted by $n_i(r)$, which represents the ith customer that is served in a particular route *r*. It must be an element in the customer set, i.e. $n_i(r) \in V$.

3) Vehicles and capacity constraint: All vehicles are identical and each one has a capacity limit *C*. This capacity limit acts as a constraint and a route failure occurs when this constraint is compromised.

4) Routes: A vehicle starts its route at the depot, visits a number of customers, and returns to the depot. A route *r* is represented as $\Omega(r) = \langle v_0, n_1(r), n_2(r), ..., n_k(r), v_0 \rangle$, where *k* is the size of the route. Since all vehicles must depart and return to the depot v_0, to simplify the representation, the depot will be omitted, i.e. $\Omega(r) = \langle n_1(r), n_2(r), ..., n_k(r) \rangle$.

5) Euclidean costs: The travel distance between any two points *i* and *j*, where each point can be a customer or the depot, is equal to the travel time and is denoted by c_{ij}, which is calculated using the following equation:

$$c_{ij} = \sqrt{(i_x - j_x)^2 + (i_y - j_y)^2} \tag{6.1}$$

where i_x and i_y are the x and y coordinates of the point i, respectively. c_{ij} is symmetrical, i.e. $c_{ij} = c_{ji}$, and satisfies the triangular inequality, where $c_{ij} + c_{jk} \geq c_{ik}$.

6) Route failure and recourse policy: For a route $\Omega(r) = \langle n_1(r), ..., n_f(r), ..., n_k(r) \rangle$, route failure is said to occur at the fth customer of the route if $\sum_{i=1}^{f} d_{n_i(r)} \geq C$ and the simple recourse policy (Gendreau *et al*., 1995, 1996b; Laporte and Louveaux, 1993, 1998; Teodorović and Pavković, 1996; Teodorović and Lucić, 2000) is employed to maintain the feasibility of solutions. For the case where $\sum_{i=1}^{f} d_{n_i(r)} > C$, the recourse policy is such that the vehicle will unload all remaining goods (equivalent to $C - \sum_{i=1}^{f-1} d_{n_i(r)}$ units) at the fth customer, return to the depot to restock, then turn back to the fth customer to complete the service, and finally continue with the originally planned route. For the case where $\sum_{i=1}^{f} d_{n_i(r)} = C$ and $f < k$, the recourse policy is such that the vehicle will return to the depot to restock and continue with the planned route at the $(f + 1)^{\text{th}}$ customer. These recourse actions will of course incur additional transportation cost, in terms of the travel distance $Q_d(r)$ and time $Q_t(r)$ for the to and fro trips to the depot. $Q_t(r)$ also includes the additional

service times incurred when a vehicle visits a customer more than once or returns to the depot for restocking due to route failures.

7) Time constraint and driver remuneration: The total duration $c_t\left(r\right)$ of a route $\Omega(r)=\left\langle n_1(r),n_2(r),...,n_k(r)\right\rangle$ is calculated as in the following equation:

$$c_t\left(r\right)=\sum_{i=1}^{k-1}\left(c_{n_i(r),n_{i+1}(r)}\right)+c_{v_0,n_1(r)}+c_{n_k(r),v_0}+\sum_{j=1}^{k}s_j+Q_t(r) \tag{6.2}$$

The time constraint is such that $c_t\left(r\right)$ should not exceed a given bound B. This is a soft constraint and B is calculated as the time for a vehicle to travel diagonally across the map from one corner to the other and back. This time is assumed to be 8 hours, equivalent to a driver's workday. Remuneration is such that drivers are paid \$10 for each of the first 8 hours of work and \$20 for every additional hour of work subsequently. This is done to penalize exceedingly long routes which may not be feasible to implement in the context of the real-world.

8) Transportation costs: The transportation costs include travel distance and driver remuneration. The travel distance $c_d\left(r\right)$ for a route $\Omega(r)=\left\langle n_1(r),n_2(r),...,n_k(r)\right\rangle$ is given in the following equation:

$$c_d\left(r\right)=\sum_{i=1}^{k-1}\left(c_{n_i(r),n_{i+1}(r)}\right)+c_{v_0,n_1(r)}+c_{n_k(r),v_0}+Q_d(r) \tag{6.3}$$

The driver remuneration $c_r(r)$ is calculated as in the following equation:

$$c_r(r) = \begin{cases} \dfrac{c_t(r)}{B/8} \times 10 & c_t(r) \le B \\ 80 + \dfrac{c_t(r) - B}{B/8} \times 20 & \text{otherwise} \end{cases} \tag{6.4}$$

9) Routing plan: The routing plan G consists of a set of routes $\{\Omega(r_1), \ldots, \Omega(r_m)\}$. The number of routes m is equal to the number of vehicles used in the plan. The condition $\bigcup_{i=1}^{m} \Omega(r_i) = V$, i.e. all customers must be routed, must be satisfied.

10) Other assumptions: It is also assumed that each customer can only be serviced by one vehicle but the vehicle is allowed to service the same customer more than once. Multiple service times will be incurred if a vehicle visits a customer multiple times.

The VRPSD, therefore, involves finding a solution $G = \{\Omega(r_1), \ldots, \Omega(r_m)\}$ that minimizes the three objectives of travel distance $\sum_{i=1}^{m} c_d(r_i)$, driver remuneration $\sum_{i=1}^{m} c_r(r_i)$, and number of vehicles required m.

6.3 Multi-Objective Evolutionary Algorithm

From the discussions in the previous sections, it is clear that the VRPSD is inherently a multi-objective problem. This section presents the multi-objective evolutionary algorithm (MOEA) specifically designed to solve the VRPSD by minimizing concurrently the three objectives of travel distance, driver remuneration, and number of vehicles required.

6.3.1 Variable-Length Chromosome

Similar to the MOEA designed to solve the exam timetabling problem in Chapter 4, a variable-length chromosome representation (Tan *et al*., 2003a, 2003b), shown in Fig. 6.2, is utilized. Each chromosome encodes a complete solution, including the number of vehicles and the customers served by these vehicles. A chromosome may consist of several routes and each route or gene is not a constant but a sequence of customers to be served. The representation allows the number of vehicles to be manipulated and minimized directly for multi-objective optimization in the VRPSD.

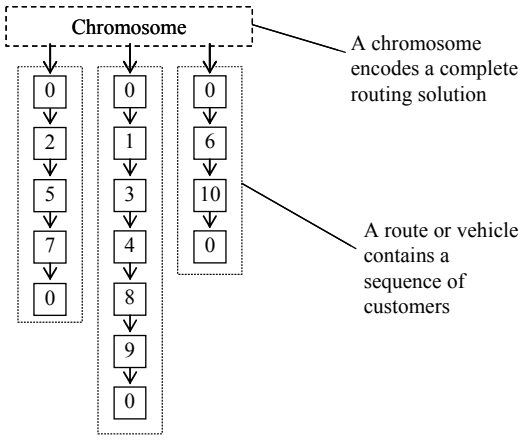


Fig. 6.2 Variable-length chromosome representation

6.3.2 Population Initialization

In the population initialization process, the first chromosome is built such that the sum of the mean values of the customer demands on each route does not exceed the vehicle capacity. Furthermore, the sum of the travel and service times on each route must not exceed the vehicle time window. The number of vehicles required in this first chromosome is then taken as the maximum number of vehicles that each of the remaining chromosomes can use. For each subsequent chromosome, the number of vehicles is randomly picked from this feasible range. The routes are then built such that each route has approximately the same number of customers. This procedure is done so that the initial population has a wide range of chromosomes with different number of vehicles to start with.

6.3.3 Route-Exchange Crossover

In the route-exchange crossover, whose operation is shown in Fig. 6.3, only the best routes of the selected chromosomes are eligible for exchange. In the case where one of the selected chromosomes has only one route, a segment of the route is randomly selected to exchange with the other chromosome's best route which will be inserted as a new route in the first chromosome. To ensure the feasibility of chromosomes after the crossover, duplicated customers are deleted. These customers are deleted from the original routes while the newly inserted route is left intact. A random shuffling operator is then applied to increase the diversity of chromosomes. In the random shuffling operation, with the exception of the newly inserted route, the order of customers in each of the remaining routes is shuffled with a probability equal to the shuffle rate.

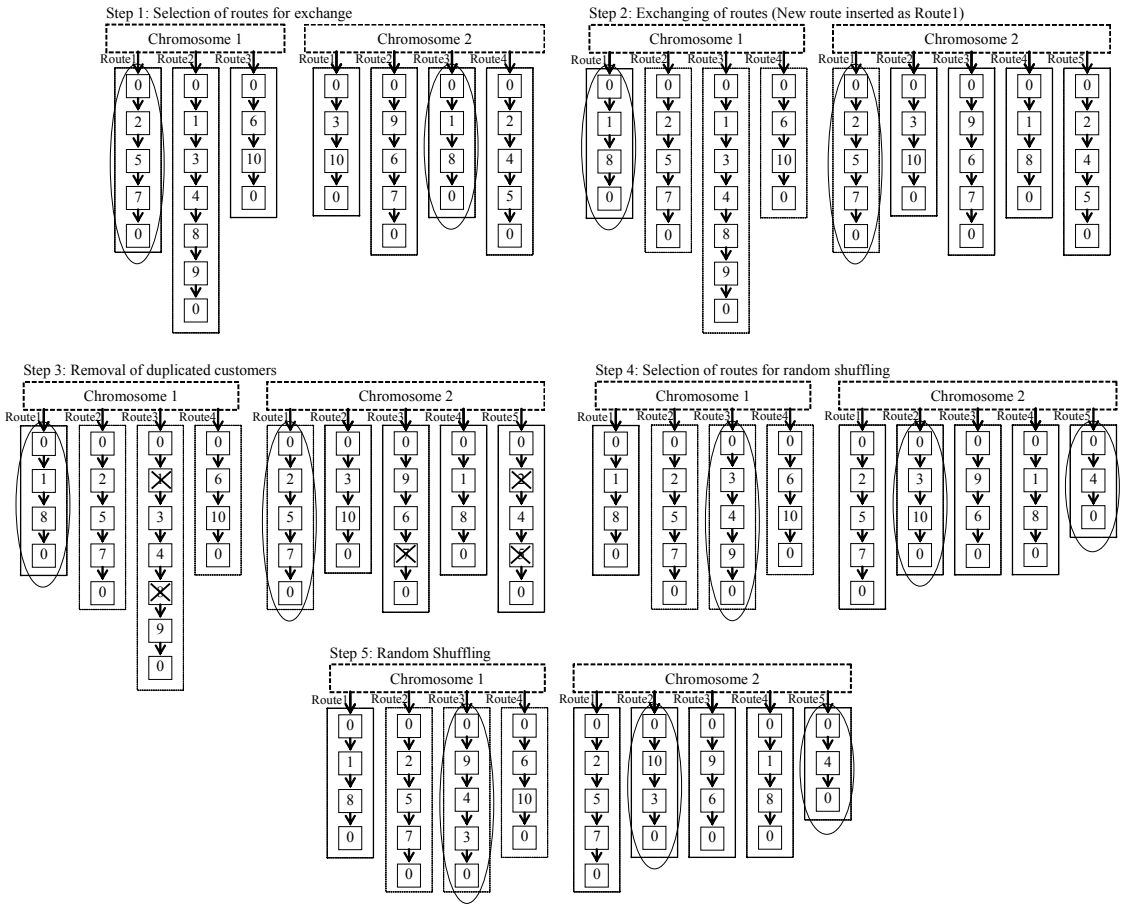


Fig. 6.3 Illustration of route-exchange crossover

6.3.4 Multi-Mode Mutation

The multi-mode mutation essentially consists of three different modes of operation and only one of the modes is applied to each chromosome selected to undergo mutation based on the mutation rate. There are three parameters associated with the mutation operator, namely elastic rate, squeeze rate, and shuffle rate. Fig. 6.4 shows the operation of the multi-mode mutation.

1) Partial swap: The partial swap operator involves a number of swap moves and for each move, two routes will be randomly chosen. A segment is then randomly

selected from each route and swapped to the other route. This new segment takes the place of the previous segment that has been swapped out. In the situation where either one of the routes has only a customer in it, a random segment is still selected from the route with more than one customer. This segment is then swapped with the solitary customer in the other route. In addition, a mechanism is in place such that the same two routes will not be selected twice in a particular partial swap operation.

2) Merge shortest route: This operation searches for the two routes of the chromosome with the smallest sum of travel distance and driver remuneration, and appends one route to the other. The merge shortest route will not operate on any chromosome with only one route.

3) Split longest route: This operation searches for the route with the largest sum of travel distance and driver remuneration, and breaks the route into two at a random point.

Like the route-exchange crossover, at the end of the multi-mode mutation, the random shuffling operation is applied on every route of each chromosome with a probability equal to the shuffle rate.

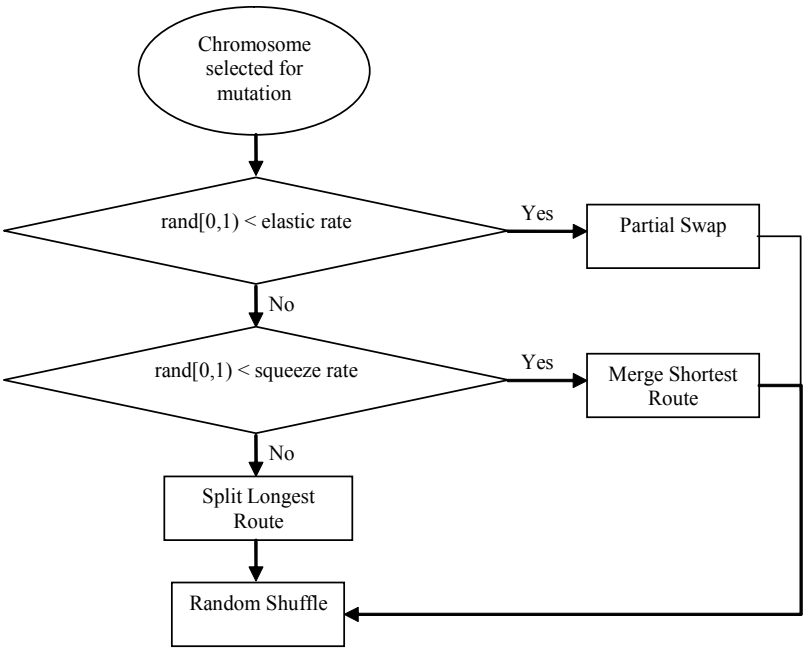


Fig. 6.4 Operation of multi-mode mutation

6.3.5 Local Search Exploitation

Two local search operators are employed in the MOEA. They are inspired by the underlying structures of a VRPSD solution identified by Dror and Trudeau (1986).

1) Shortest path search: The shortest path search (SPS) is designed to exploit the fact that route failures are more likely to occur at the end of a route. The SPS attempts to rearrange the order of customers in a particular route. For example, given a route that contains five customers, a new route is built by choosing the customer that is furthest from the depot as the first customer in the route, while the customer that is nearest to the depot is chosen as the last customer of the route. Next, the customer that is nearest to the first customer is chosen as the second customer, while the customer that is nearest to the last customer is chosen as the second last customer

of the new route. This step continues until all the customers in the original route are re-routed. The new route will be compared against the original one and the better route will be retained. By re-routing customers in such a manner, customers that are further from the depot will be at the beginning of the route whereas those that are nearer to the depot will be at the end of the route. The rationale is to reduce the additional transportation cost that will be incurred by the recourse policy.

2) Which directional search: The which directional search (WDS) is designed to exploit the fact that the expected transportation cost of a route is dependent on the traversed direction. In contrast, for the deterministic VRP, the transportation cost of a route is the same regardless of the direction in which the route is traversed. To be specific, given a route, the WDS builds a new route that runs in the opposite direction. Similarly, the new route will replace the original one if it is better.

6.3.6 Route Simulation Method

As mentioned earlier, one of the main difficulties of solving the VRPSD is in finding an objective function that is able to define properly the expected transportation cost of a solution. In this section, the route simulation method (RSM) is proposed to evaluate the expected costs of solutions. The fundamental idea behind the RSM is based on the sampling method of Lee and Chew (2003), who applied the method for numerical optimization. Fig. 6.5 will be used to illustrate the operation of the RSM.

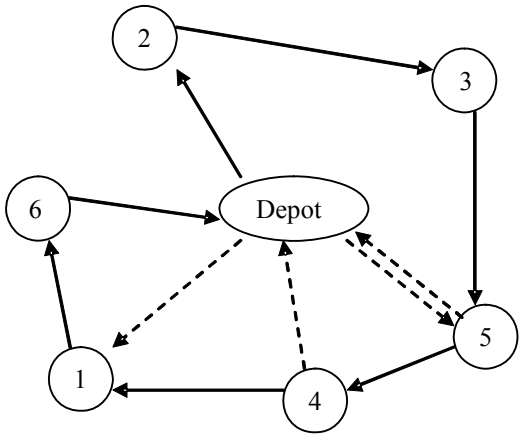
Fig. 6.5 shows a route sequence, Depot → 2 → 3 → 5 → 4 → 1 → 6 → Depot. The solid arrows indicate the route that the vehicle will take if this were a

deterministic VRP. In the VRPSD, due to the recourse policies in the event of a route failure, the actual route taken by the vehicle cannot be known with certainty before the route is actually implemented. However, the implementation of the route can be simulated by generating a set of demands of all the customers based on their demand distributions and treating these demands as if they were the real demands revealed when a vehicle first arrives at the customer. The set of demands generated is tabulated in Fig. 6.5. For this particular example, it is assumed that the vehicle capacity is 15 and each arrow indicates a unit of distance.

The vehicle first leaves the depot and arrives at customer 2. It is able to satisfy its demand with a remaining capacity of 9. The vehicle then travels to customer 3 and satisfies its demand. The capacity of the vehicle is 7 when it reaches customer 5. The vehicle then finds that it is unable to satisfy the demand of customer 5, so it unloads all remaining goods and makes a return trip to the depot to restock. This recourse is indicated by the dashed arrows between the depot and customer 5. The vehicle then unloads two units of goods and leaves customer 5 for customer 4 with a capacity of 13. After serving customer 4, the vehicle is empty and returns to the depot to restock. Since the demand of customer 4 has been satisfied, the vehicle travels to customer 1 from the depot. The vehicle then satisfies the demands of customers 1 and 6 and returns to the depot. From this simulation, the total distance traveled by the vehicle (10 units for this example) and the remuneration for the driver for a particular realization of the set of customer demands can be obtained.

Due to the stochastic nature of the cost considered, there is a need to repeat the above operation N times for every route of a particular solution, using a different set

of demands randomly generated based on the demand distributions of the customers each time and then taking the average to obtain the expected transportation cost of the solution. In this chapter, three different RSM settings are studied. Generate every generation (GEG) refers to the setting where the *N* demand sets, which are used by the RSM, are refreshed every generation. Generate every *M* (GEM) refers to the setting where the *N* demand sets are only refreshed at the end of every *M* generations. Lastly, alternate every *M* (AEM) refers to the setting where for *M* generations, the RSM uses the *N* randomly generated demand sets and for the next *M* generations, the RSM uses the mean values of the customers' demand distributions to simulate the implementations of the routes. The *N* demand sets are then refreshed for use over the next *M* generations and the process repeats.



Customer	Real demand
1	5
2	6
3	2
4	13
5	9
6	5

Fig. 6.5 Example to show the operation of the RSM

6.3.7 Computing Budget

The computing budget (Chen *et al*., 1997; Lee and Chew, 2003) represents a fixed amount of computational work. As can be observed from the previous section, the RSM requires intensive computations and can be regarded as the bottleneck of the whole algorithm. As such, it makes sense to define one unit of the computing budget as one run of the RSM on a particular solution using a particular demand set. Thus, the computing budget puts a cap on the total number of times the RSM is run on solutions in the MOEA and can be regarded as the stopping criterion of the evolutionary algorithm. For GEG and GEM, the computing budget set is actually approximately equal to the product of the size of the search population, the number of generations used in the MOEA, and *N*, the number of times the RSM is repeated to obtain the expected transportation cost of a solution. This product gives only an approximate value due to the additional runs of the RSM during local search. As for AEM, the following equation applies:

$$Computing\ Budget \approx 0.5 \cdot \left(GEN \cdot POPSIZE\right) \cdot \left(N + 1\right) \tag{6.5}$$

where *GEN* is the number of generations and *POPSIZE* is the population size.

This equation takes into account the fact that AEM alternates between using *N* randomly generated demand sets and the mean demand set for the RSM every *M* generations. As such, the RSM needs to run *N* times when using the *N* randomly generated demand sets but only runs one time when using the mean demand set. The

equation only gives an approximation, again due to the additional runs of the RSM during local search.

6.4 Simulation Results and Analysis

The MOEA was programmed in C++ and simulations were performed on an Intel Pentium 4 2.8 GHz computer. Table 6.1 shows the parameter settings chosen after some preliminary experiments.

Table 6.1 Parameter settings for simulation study

Parameter	Value
Population size	500
Crossover rate	0.7
Mutation rate	0.4
Elastic rate	0.5
Squeeze rate	0.5
Shuffle rate	0.3
Computing budget	2,000,000

Bianchi *et al*. (2004) highlighted that there is no commonly used benchmark for the VRPSD in the literature. As such, many authors generated their own test problems. Teodorović and Lucić (2000) and Yang *et al*. (2000) randomly generated the locations of the depot and the customers. The characteristics of each customer's demand, mean and variance, are also randomly generated. Dror and Trudeau (1986)

adapted a deterministic VRP test problem for the VRPSD. The test problem is constructed by adding a demand standard deviation to each customer. The original demand quantity is used as the mean demand of each customer. The standard deviation of the demand of each customer is generated using a uniform random number generator so that it falls between zero and one-third of the mean demand of the customer. On the other hand, Bianchi *et al*. (2004, 2006) did not choose the locations of the customers uniformly at random, but randomly with normal distributions around two centers so that the customers are grouped into two clusters. As Dror and Trudeau (1986) are the only one that provided the actual test problem used, in developing the MOEA, all simulations, unless otherwise stated, were performed using that test problem. The test problem will be referred to as DT86. It is to be noted that all the customers' demands in DT86 are normally distributed. Since the time constrained problem was not considered by Dror and Trudeau (1986), the service time t_s of each customer is filled in by subtracting 10 from the mean demand or one unit, whichever is larger. This is done so that a customer with a higher mean demand would require a longer service time. A fixed service time of 10 is also set for the depot. Since DT86 uses a 70 × 80 map, the vehicle time window is calculated to be 212 units. This time window is equivalent to 8 hours. As such, each hour corresponds to 26.5 units, which is used to compute the remuneration for drivers according to the rates given in Section 6.2.2.

The subsequent sections present the extensive simulation results and analysis of the proposed MOEA. Section 6.4.1 demonstrates the effectiveness of the proposed

hybrid local search, as well as analyzes how the various settings in which the local search heuristics are incorporated with the MOEA will affect the performance of the algorithm. Section 6.4.2 demonstrates the advantages of multi-objective optimization and at the same time shows the relationships between the three objectives of travel distance, driver remuneration, and number of vehicles required. Section 6.4.3 presents a new way of assessing the quality of solutions to the VRPSD on top of comparing their expected transportation costs and shows that the MOEA, equipped with the RSM, is able to produce solutions that are robust to the stochastic nature of the problem. Section 6.4.4 shows how the value of the parameter N affects the performance of GEG, whereas Section 6.4.5 attempts to study how the value of the parameter M affects the performance of GEM. Section 6.4.6 tests the performance of the MOEA on three VRPSD instances adapted from Solomon's vehicle routing problem with time windows (VRPTW) benchmark problems (Solomon, 1987). Section 6.4.7 discusses how the RSM can actually be implemented in practice.

6.4.1 Performance of Hybrid Local Search

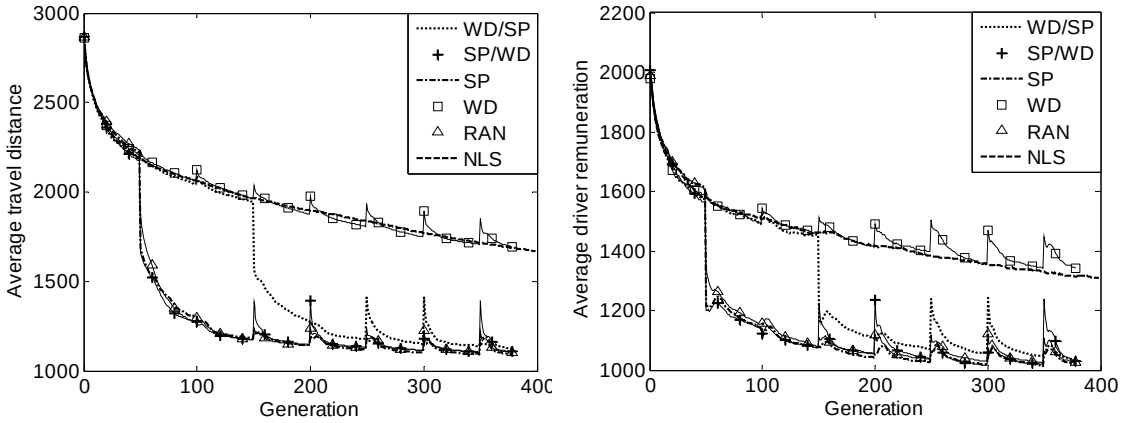
The MOEA incorporates the local search heuristics in order to exploit local routing solutions in parallel with global evolutionary optimization. The local search heuristics are specially designed to exploit the route structures of a solution to the VRPSD. This section demonstrates the effectiveness of local exploitation in the MOEA and also analyzes the effectiveness of various settings in which the local search heuristics are incorporated with the MOEA.

Simulations were conducted using six different settings. Three of the settings include the MOEA with no local search (NLS), with only the WDS (WD), and with only the SPS (SP). The local search operators are applied to the evolving and archive populations every 50 generations. WD/SP is a setting which involves the application of the WDS for the first two local exploitations in the MOEA, i.e. on the 50th and the 100th generation, and alternates between the two local search heuristics every 100 generations, while applying local search every 50 generations. On the other hand, SP/WD starts with the SPS on the 50th and the 100th generation, and alternates between the two local search heuristics every 100 generations. The final setting is RAN, where during the application of local search every 50 generations of the MOEA, each chromosome will have equal chance of being applied by either the SPS or the WDS on all of its routes. Each of the six settings underwent 10 simulation runs. The simulations were conducted using the GEG setting with N, the number of times the RSM is repeated to obtain the expected transportation cost of a solution, set to 10.

The convergence traces of the travel distance and the driver remuneration for the six settings are plotted in Fig. 6.6(a)-(b) and Fig. 6.7(a)-(b). Fig. 6.6(a)-(b) shows the convergence of the respective costs, averaged over all the solutions in the archive population, over the generations. Fig. 6.7(a)-(b) shows the same costs averaged over all the non-dominated solutions in the archive population. The costs are further averaged over the 10 simulation runs performed. The plots show the effectiveness of local exploitation in the MOEA as the five settings which use local search perform better than NLS. The effectiveness of the SPS is evident since the four settings, namely WD/SP, SP/WD, SP, and RAN, which make use of the local search operator,

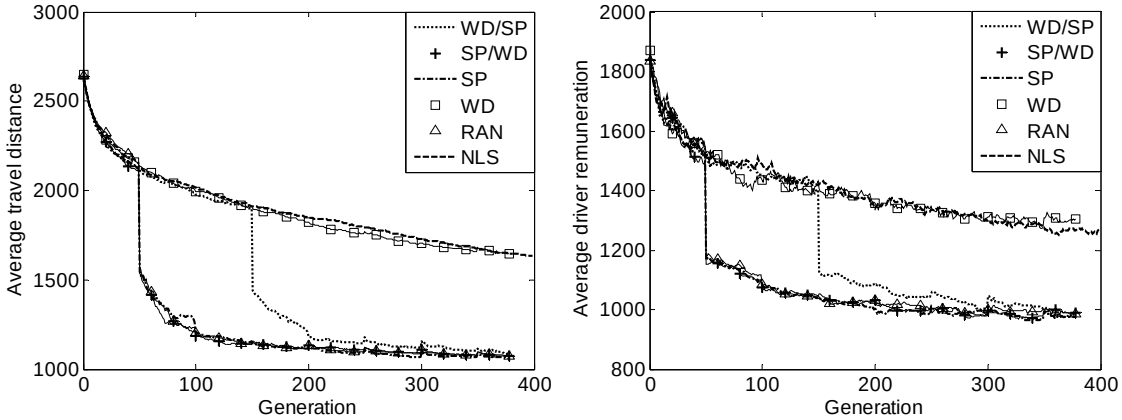
are able to find solutions with travel distance and driver remuneration significantly lower than those found by WD and NLS. The SPS is able to speed up convergence as it causes sharp dips in the respective costs of the solutions found whenever it is performed. The performances of WD/SP, SP/WD, SP, and RAN are comparable and the setting WD/SP is selected as the default setting for any further analysis unless otherwise stated.

In Fig. 6.6(a)-(b), it is observed that there are distinctive spikes in the convergence traces which coincide with the occurrences of local search. This is despite the fact that during local search, a new route is constructed and compared with the original one and the better route is retained. This happens because in comparing the new and original routes during local search, the solutions in the archive population are re-evaluated by the RSM. This re-evaluation acts to complement the RSM and is important in the stochastic problem where the costs of solutions are sensitive to the demand sets that are used by the RSM. For a particular solution, the fitness evaluated using the RSM can be very different depending on the demand sets used. As such, it is essential that a solution to the VRPSD be robust to the stochastic nature of the problem and its fitness should not differ too much with each evaluation by the RSM. The re-evaluation of all the solutions in the archive population during local search ensures that only robust solutions stay non-dominated. The effect of this can be seen in Fig. 6.7(a)-(b) which considers only non-dominated solutions in the archive population. The spikes in these convergence traces during local search are significantly smaller, if not negligible.



(a) (b)

Fig. 6.6 (a) Average travel distance and (b) average driver remuneration of archive populations for different local search settings

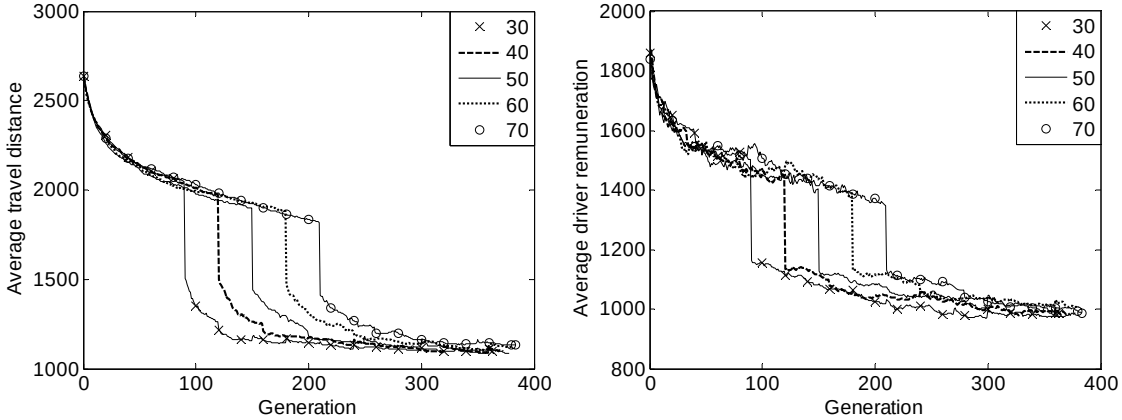


(a) (b)

Fig. 6.7 (a) Average travel distance and (b) average driver remuneration of non-dominated solutions for different local search settings

Another test was conducted to study if the frequency at which local search is performed has any effect on the performance of the MOEA. On top of the setting of applying local search every 50 generations, four other settings where local search is applied every 30, 40, 60, and 70 generations, respectively, were used in the test. Ten simulation runs of each of the five settings were performed.

The convergence traces of the travel distance and the driver remuneration, averaged over the non-dominated solutions in the archive population and over the 10 simulation runs, for the five settings are plotted in Fig. 6.8(a)-(b), respectively. The plots show that the performances of the five settings are comparable. It is to be noted that with more frequent application of local search, the number of generations used in the MOEA is reduced. This is because additional runs of the RSM are required during local search to evaluate the new routes constructed by the local search heuristics. With a fixed computing budget, the additional RSM runs translate to a smaller number of generations used in the MOEA. All further simulation runs are conducted using the setting where local search is applied every 50 generations.



(a) (b)

Fig. 6.8 (a) Average travel distance and (b) average driver remuneration of non-dominated solutions for different local search generations

6.4.2 Multi-Objective Optimization Performance

This section presents the routing performance of the MOEA, particularly on its multi-objective optimization that offers the advantages of improved routing solutions and the exploration of a larger search space. The relationships between the three objectives of travel distance, driver remuneration, and number of vehicles required will also be shown.

To illustrate the multi-objective optimization performance of the MOEA, seven types of simulations, with similar settings but different sets of optimization criteria (for evolutionary selection operation), were performed. Three of the simulation types are concerned with minimizing the single objectives of travel distance (SOD), driver remuneration (SOR), and number of vehicles (SOV), respectively, while another three are concerned with minimizing two objectives concurrently, namely, travel distance and driver remuneration (DODR), travel distance and number of vehicles (DODV), and driver remuneration and number of vehicles (DORV). The final simulation type optimizes the three objectives concurrently (MO). Ten simulation runs of each of the simulation types were performed.

Fig. 6.9 shows the comparison results for the evolutionary optimization of the seven simulation types. The comparisons were performed using the multiplicative aggregation method (Van Veldhuizen and Lamont, 1998) of travel distance and driver remuneration averaged over all solutions and all non-dominated solutions, respectively, in the archive population found at the termination of the algorithm. The results are further averaged over the 10 simulation runs of each simulation type that

were performed. As can be seen from Fig. 6.9, MO produces the best performance with the smallest product of travel distance and driver remuneration.

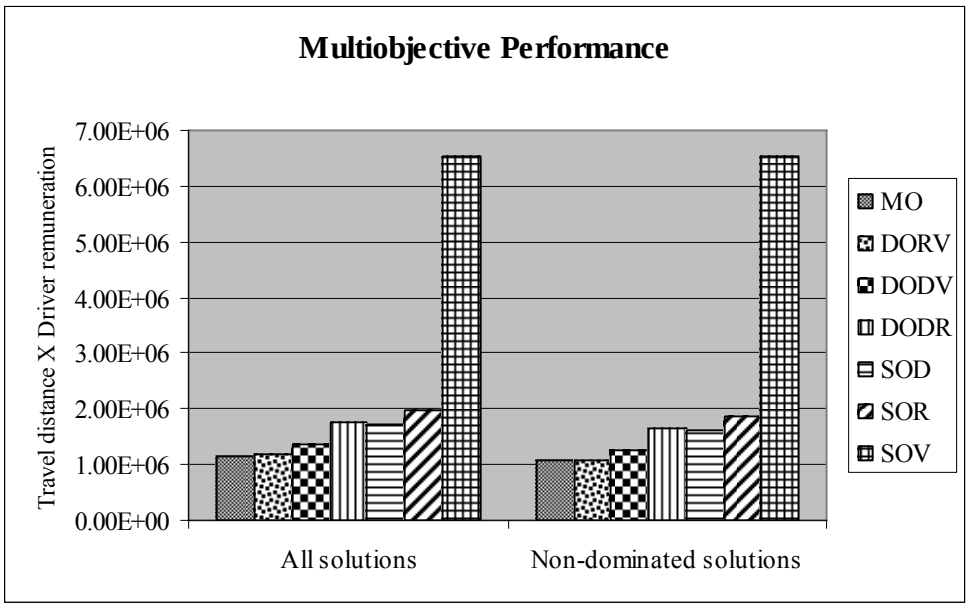
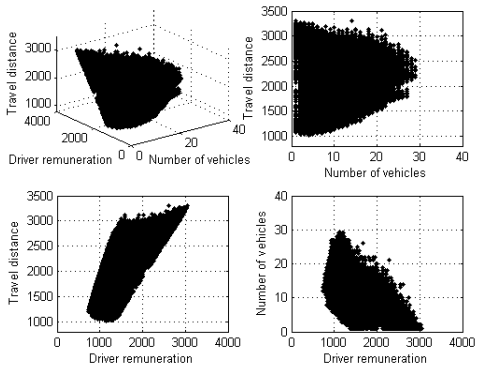


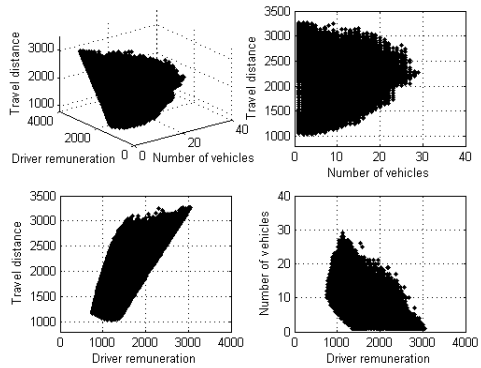
Fig. 6.9 Performance comparison for different optimization criteria

The search spaces in the objective domain explored by the seven simulation types are also plotted in Fig. 6.10(a)-(g). Each point in the plots is a point in the objective domain that has been found by the respective simulation types during the algorithm. Separate two-dimensional graphs are also plotted for clarity in analyzing the size of the search spaces explored by the seven simulation types. As can be seen from the plots, with the exception of DORV, the search space explored by MO is considerably larger than those explored by the remaining five simulation types. Most of these five simulation types are trapped in some local optima. For example, from Fig. 6.10(d)-(f), DODR, SOD, and SOR, respectively, are unable to find any solution

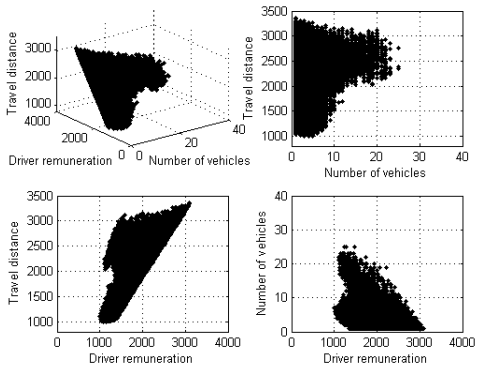
with travel distance below 1500, whereas from Fig. 6.10(g), SOV is unable to find any solution with driver remuneration below 1000. All these show that there is a need to use MO to minimize the three objectives concurrently in the multi-objective VRPSD. By using MO, the diversity of points found is increased, allowing it to escape from local optima and explore a larger search space. These advantages translate to better routing solutions.



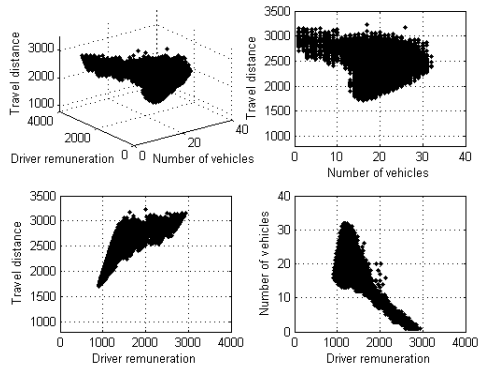
(a) MO



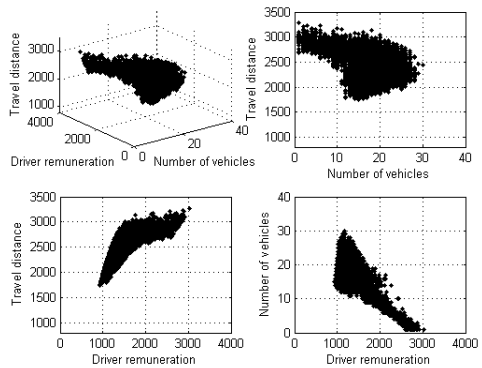
(b) DORV



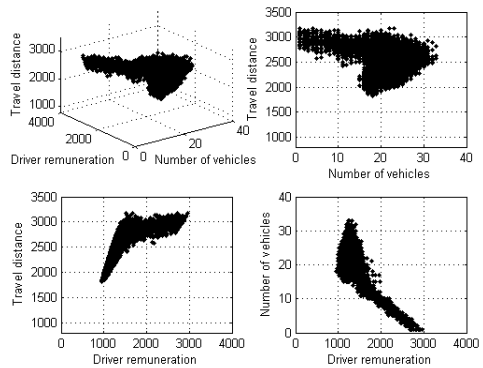
(c) DODV



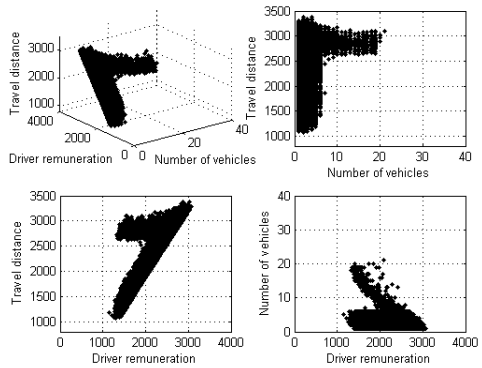
(d) DODR



(e) SOD



(f) SOR



(g) SOV

Fig. 6.10 Comparison of search spaces for different optimization criteria

It has been shown that there is a need to use MO to minimize concurrently the three objectives of travel distance, driver remuneration, and number of vehicles required. Although the three objectives are quantitatively measurable, the relationships between them in a routing problem are unknown until the problem has been solved. The objectives may be positively correlated to each other, or they may be conflicting with each other. To see how the objectives are related to one another, Fig. 6.10(a), the search space of MO, is magnified to show the essential information and plotted in Fig. 6.11.

From the graph of travel distance against driver remuneration in Fig. 6.11, it can be observed that the two objectives are conflicting with each other, i.e. any attempt to minimize either of the objectives will cause the other objective to increase. This is also the case for the number of vehicles required and driver remuneration. On the other hand, from the graph of travel distance against number of vehicles, it is observed that the two objectives are correlated to each other, i.e. it is possible to minimize both concurrently.

The above observations are consistent with the studies of Yang *et al*. (2000) who showed that if time constraints are not considered, the optimal route, in terms of travel distance, is always a single route. The results also show that the single route solution, which has the lowest travel distance, may not be feasible to implement in the context of the real-world as a low travel distance corresponds to a high driver remuneration which translates into the driver working deep into overtime.

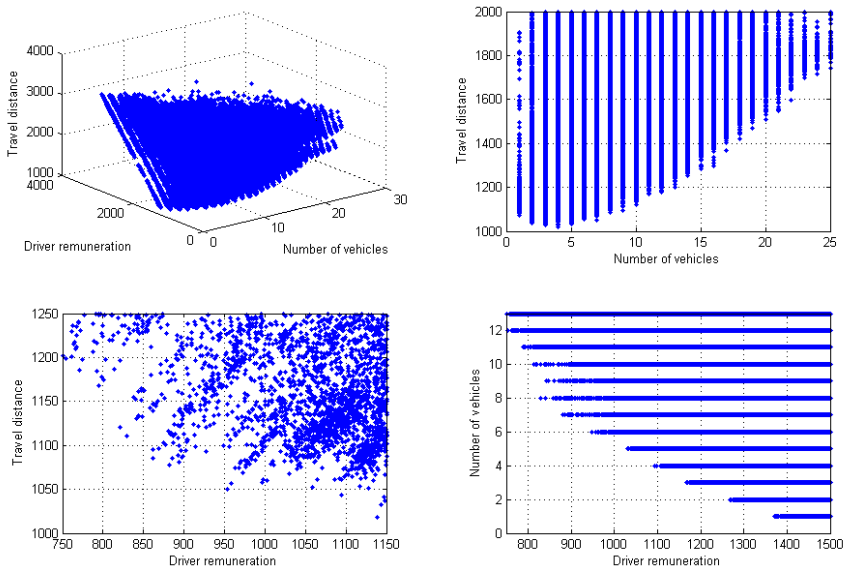


Fig. 6.11 Magnified search space of MO

6.4.3 Comparison with a Deterministic Approach

In the absence of a stochastic procedure to deal with stochastic demands, one can generate the routes using a deterministic vehicle routing algorithm by treating the expected demand at each customer as its deterministic demand (Yang *et al*., 2000). The attraction of this deterministic approach is its relative simplicity and familiarity to practitioners. The MOEA can in fact be modified into a deterministic vehicle routing algorithm by solely using the mean demand set in the RSM. However, what makes the MOEA different from a deterministic vehicle routing algorithm is the RSM's ability to operate on demand sets which are randomly generated based on the demand distributions. This section will show that the RSM's ability to operate on randomly generated demand sets can lead to solutions which are more robust to the stochastic nature of the problem compared to the deterministic approach and that the

expected transportation costs of such solutions are good estimates of the true performance of the solutions. In addition, a RSM-based procedure is proposed to assess the quality of solutions on top of comparing their expected costs.

For comparison purposes, simulations were conducted on the three RSM settings, GEG, GEM, and AEM, which were introduced in Section 6.3.6, with N and M both set to 10. The results of these settings are compared with the deterministic approach (DET) mentioned in the previous paragraph. These four settings provide a spectrum, from pure stochastic to pure deterministic, of approaches to the VRPSD. Ten simulation runs of each of the four settings were performed.

The convergence traces of the travel distance and the driver remuneration, averaged over the non-dominated solutions and over the 10 simulation runs, for the four settings are plotted in Fig. 6.12(a)-(b). Due to the nature with which the RSM is run in AEM and DET, i.e. the RSM is run only once, instead of N times, when evaluating solutions using the mean demand set, the MOEA for these two settings took more than 500 generations to complete. However, by the 500th generation, these two settings have converged and the plots in Fig. 6.12(a)-(b) show only the convergence traces up to the 500th generation. By comparing the convergence traces of the four settings, it appears that DET is able to churn out the best solutions since both the average travel distance and the average driver remuneration are the lowest among the four settings at the termination of the algorithm.

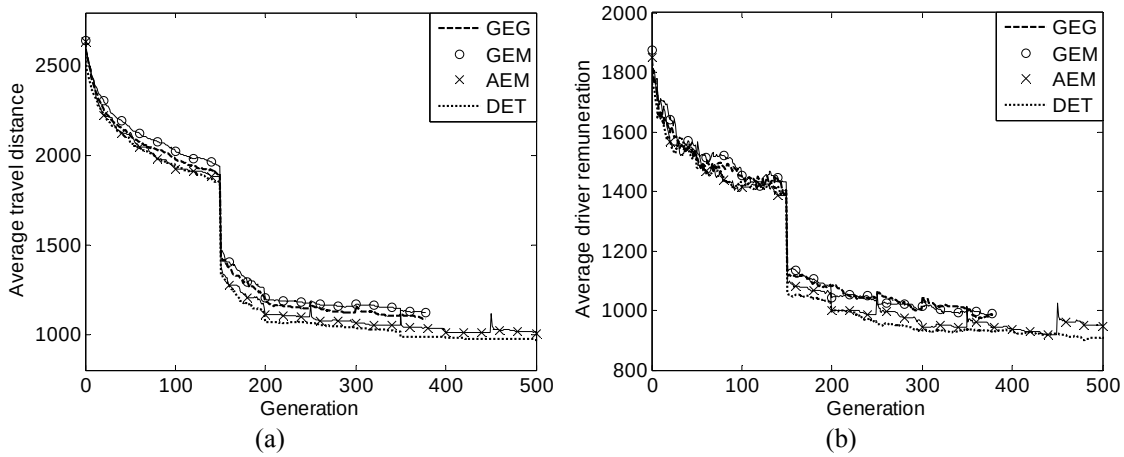


Fig. 6.12 (a) Average travel distance and (b) average driver remuneration of non-dominated solutions of GEG, GEM, AEM, and DET

It was highlighted in Section 6.1 that due to the stochastic nature of the problem, the actual cost of a particular solution to the VRPSD cannot be known with certainty before the actual implementation of the solution. During the decision making process, the logistic manager will look at the expected transportation costs of all the candidate solutions and choose the solution that best suits the company's logistic condition, in terms of the available vehicle fleet size, and the company's priorities of whether to take the solution with a shorter travel distance but is likely to incur greater cost in the form of the remuneration for the drivers. In view of this, for the logistic manager to make correct decisions, it is important for the expected cost of each solution to give a good estimate of the true performance of the solution, i.e. the actual cost of implementing the solution should not deviate too much from the expected cost. As such, it is necessary to compare the results to the VRPSD based on this aspect on top of comparing their expected costs.

To perform such a comparison, a test demand set is randomly generated based on the customers' demand distributions. This test demand set will represent the real demands that the vehicles of a particular solution would experience when the solution is implemented. The RSM is then operated, using that test demand set, on all the Pareto solutions found at the termination of each of the four settings, GEG, GEM, AEM, and DET, to simulate the actual costs of implementing the solutions. The deviation between the actual and expected costs of each solution is then calculated using the following equation:

$$Dev = \sqrt{\left(Dist_{Ex} - Dist_{Act}\right)^2 + \left(Rem_{Ex} - Rem_{Act}\right)^2} \tag{6.6}$$

This deviation is essentially the Euclidean distance in the objective domain between the actual and expected costs of each solution. To ensure that the results are not biased towards any test demand set, the same process is repeated for three other randomly generated test demand sets. The above procedures are repeated for the Pareto solutions found by the 10 simulation runs of each setting that were performed. The results of these comparisons are represented in box plots and are shown in Fig. 6.13. Each box plot represents the distribution of the deviations between the actual and expected results.

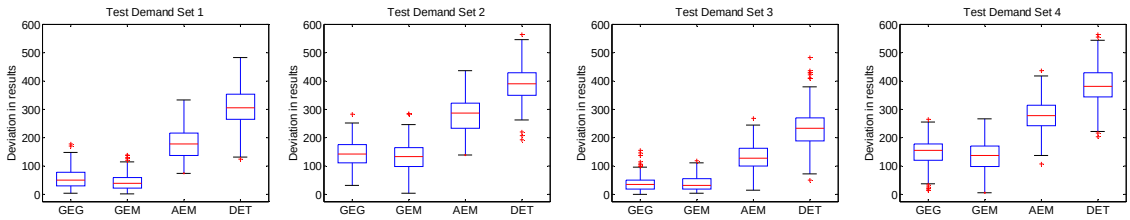


Fig. 6.13 Deviation between actual and expected costs of Pareto solutions of GEG, GEM, AEM, and DET for four test demand sets

It can be seen from Fig. 6.13 that the expected costs of solutions obtained by GEG and GEM deviate less from the corresponding actual costs for all the test demand sets compared to the other two settings. AEM and DET produce solutions that have expected costs that are poor estimates of the actual costs. The spreads of their deviations are also larger compared to those of GEG and GEM, which will result in poorer predictability in the deviations. It is noted that test demand sets 2 and 4 resulted in greater deviations for all the four settings but the order of performances of the four settings remains the same.

Although the above results show the magnitudes of deviations between the actual and expected costs of solutions, they do not show the direction of these deviations. The actual cost of a particular solution can be better than the expected cost of the solution even though the deviation between the two costs is large. To compare the performances of the four settings based on this aspect, two separate comparisons were made. The first involves comparing the increase in travel distance, from the expected value, after implementing a particular solution, whereas the other compares the increase in driver remuneration. The results of these two comparisons are shown in Fig. 6.14(a)-(b). The figures again show the same pattern where GEG and GEM

produced the most robust solutions which have expected costs that are good approximations of the actual costs.

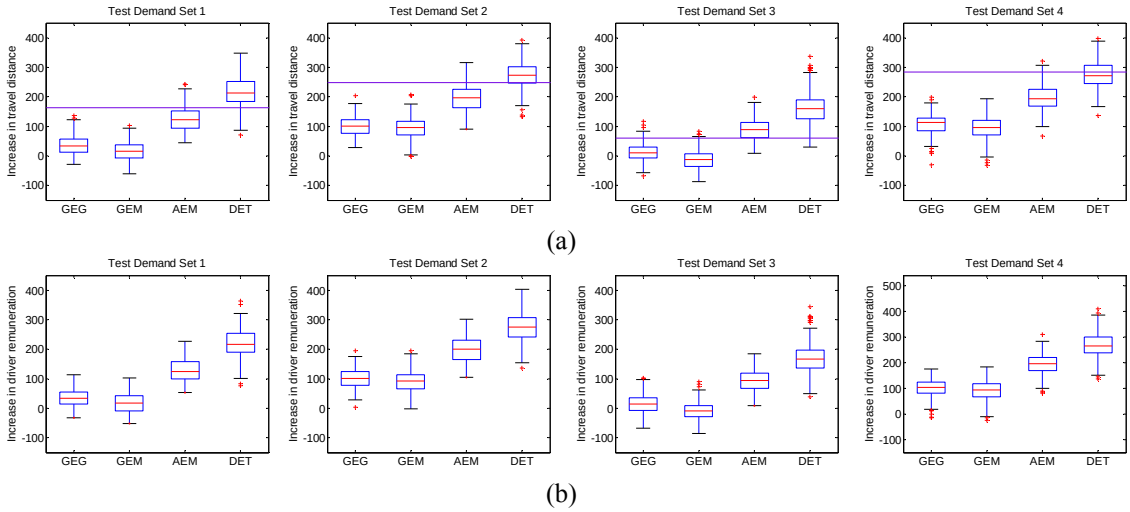


Fig. 6.14 Increase in (a) travel distance and (b) driver remuneration after implementing Pareto solutions of GEG, GEM, AEM, and DET

A test was also conducted to compare the robustness of the solution found by Dror and Trudeau (1986) with those found by the four settings. The solution of Dror and Trudeau (1986) was implemented using the simple recourse policy described in Section 6.2.2. The increases in travel distances for the four test demand sets are obtained and plotted as four horizontal lines in the respective box plots in Fig. 6.14(a) since Dror and Trudeau (1986) only considered the single objective of travel distance. From Fig. 6.14(a), it can be seen that the solution of Dror and Trudeau (1986) is not as robust as those found by GEG and GEM. For test demand sets 2 and 4, the increases in distances after implementing the solution are comparable with those found by DET. This is despite the fact that Dror and Trudeau (1986) used a worst

case recourse policy where in case of a route failure, all the remaining customers in the route are served through individual deliveries.

Table 6.2 and Table 6.3 summarize the importance of this analysis. In Table 6.2, expected travel distance and expected driver remuneration are the respective costs averaged over the Pareto solutions in the archive population at the termination of the algorithm of the four different settings and over the 10 simulation runs as plotted in Fig. 6.12(a)-(b). Increase in travel distance and increase in driver remuneration are the median values of the increase in the respective costs after implementation using test demand set 1 as plotted in Fig. 6.14(a)-(b). Actual travel distance and actual driver remuneration are the sum of the respective expected costs and increase in costs for each setting. Multiplicative aggregate (Van Veldhuizen and Lamont, 1998) shows the product of actual travel distance and actual driver remuneration.

It was previously commented that DET produces the "best" solutions among the four settings. This is reflected in Table 6.2 as DET produces solutions with the lowest expected travel distance and expected driver remuneration. However, DET also produces solutions with the largest increase in travel distance and driver remuneration after implementation using test demand set 1. Taking all these factors into consideration, Table 6.2 shows that GEG has the lowest multiplicative aggregate of 1.142, distinctly lower than the values for AEM and DET. The same analysis is also performed on the other three test demand sets (td) and the results are summarized in Table 6.3. Expected multiplicative aggregate is the product of expected travel distance and expected driver remuneration in Table 6.2 and is the same value for a particular setting regardless of the test demand set. Actual multiplicative aggregate is

equivalent to the multiplicative aggregate field in Table 6.2, except that it shows the values of the aggregate obtained considering all the four test demand sets. Average actual multiplicative aggregate is the average value of the four actual multiplicative aggregates and will be taken as the overall performance indicator as it takes into account the contributions from the four test demand sets. Like Table 6.2, Table 6.3 shows that although DET has the lowest expected multiplicative aggregate, if one were to take into account the increases in the transportation costs after implementing the solutions, the performance of DET is the worst among the four settings since it has the highest average actual multiplicative aggregate.

From this analysis, it is evident that DET is prone to giving the logistics manager inaccurate information in terms of the expected transportation cost. In order for the logistic manager to correctly select a solution to implement, it is important that the expected costs of solutions should give good approximations of the actual costs. This analysis shows that in assessing the quality of solutions, comparing their expected costs is not enough. This study also shows that the stochastic nature of the VRPSD cannot be neglected and that the RSM using demand sets randomly generated based on the customers' demand distributions is a robust technique to evaluate the expected cost of a solution, which was previously considered in the literature as one of the main difficulties to solving the VRPSD.

Table 6.2 Comparison with a deterministic approach considering test demand set 1

	Expected travel distance	Increase in travel distance	Actual travel distance	Expected driver remuneration	Increase in driver remuneration	Actual driver remuneration	Multiplicative aggregate ($\times 10^6$)
GEG	1086.59	33.85	1120.44	985.953	32.874	1018.83	1.142
GEM	1120.52	16.535	1137.06	990.334	18.639	1008.97	1.147
AEM	1002.08	122.35	1124.43	947.039	125.044	1072.08	1.205
DET	970.174	213.3955	1183.57	909.587	217.347	1126.93	1.334

Table 6.3 Comparison with a deterministic approach considering all four test demand sets

	Expected multiplicative aggregate ($\times 10^6$)	Actual multiplicative aggregate ($\times 10^6$)				Average actual multiplicative aggregate ($\times 10^6$)
		td1	td2	td3	td4	
GEG	1.071	1.142	1.292	1.097	1.307	1.210
GEM	1.110	1.147	1.316	1.087	1.320	1.217
AEM	0.949	1.205	1.376	1.136	1.365	1.271
DET	0.882	1.334	1.475	1.216	1.460	1.371

6.4.4 Choice of *N*

N, the number of times the RSM is repeated to obtain the expected transportation cost of a solution, was set to 10 in the previous section. This section analyzes the effect of the value of *N* on the performance of the MOEA using the GEG setting in terms of the expected costs of solutions found and how well these expected costs approximate the actual costs of implementation.

Ten simulation runs of seven settings with *N* set to 1, 3, 5, 10, 30, 50, and 70, respectively, were performed. The RAN local search setting described in Section

6.4.1 is used in all the simulations here to allow a fair comparison since the *N* settings of 30, 50, and 70 run for less than 150 generations and would not be operated by the SPS if the default local search setting of WD/SP were used. The convergence traces of the travel distance and the driver remuneration, averaged over the non-dominated solutions in the archive population and over the 10 simulation runs, for the seven settings are plotted in Fig. 6.15(a)-(b). Due to space constraints, the convergence traces for the *N* settings of 1, 3, and 5 are shown only up to the 400th generation.

From Fig. 6.15(a)-(b), it can be observed that as the value of *N* increases, the number of generations used in the MOEA decreases (A portion of the convergence traces has been enlarged to highlight this point). This is because as *N* increases, more runs of the RSM is applied each time the fitness of a chromosome is evaluated and since the total number of times the RSM is applied throughout the algorithm for the seven settings is fixed at the computing budget, the number of generations used in the MOEA is reduced accordingly. This reduction in the number of generations used in the MOEA results in poorer routing solutions as the MOEA does not have sufficient time to explore the search space.

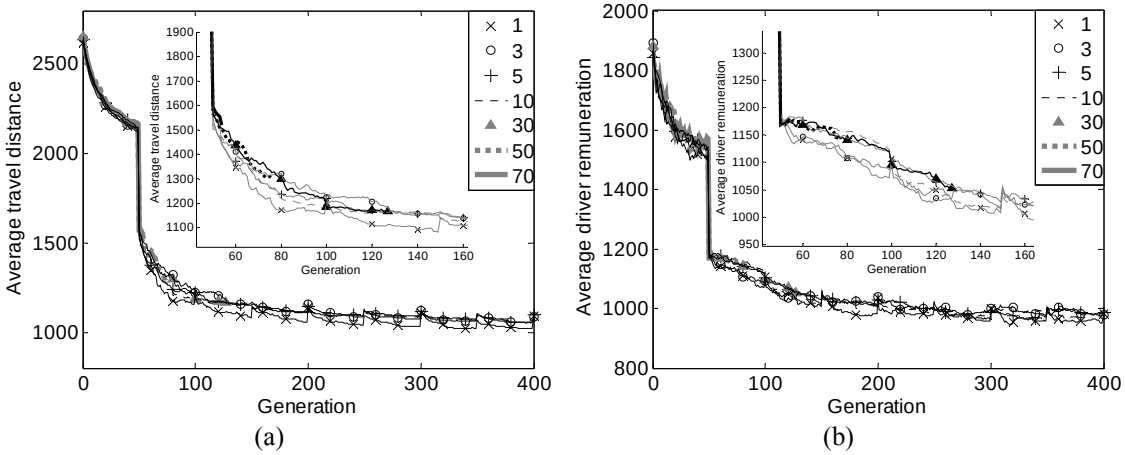


Fig. 6.15 (a) Average travel distance and (b) average driver remuneration of non-dominated solutions of GEG using different *N* values

The box plots that were used in the previous section to analyze how well the expected cost of a solution approximate the actual cost of implementing the solution are also plotted for the seven settings in Fig. 6.16(a)-(b). From Fig. 6.16(a)-(b), it can be observed that as the value of *N* increases, the solutions found are more robust to the stochastic nature of the problem in that the expected costs of the solutions are better estimates of the actual costs.

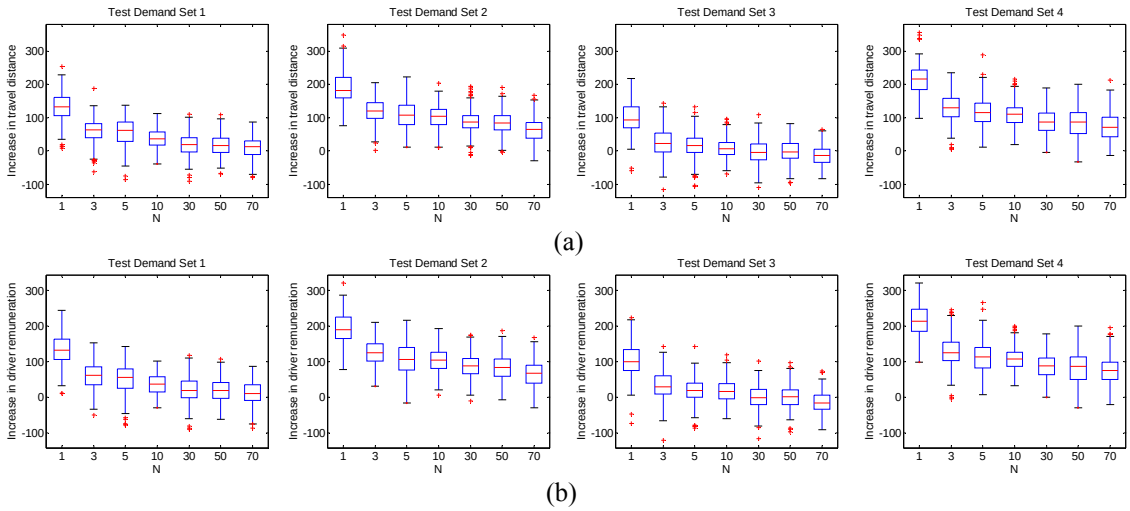


Fig. 6.16 Increase in (a) travel distance and (b) driver remuneration after implementing Pareto solutions of GEG using different *N* values

From the above results, it can be seen that while setting a larger value of *N* for GEG will produce more robust solutions whose expected costs are better approximations of the actual costs of implementation, due to the fixed computing budget, the corresponding smaller number of generations used in the MOEA will result in poorer routing solutions as there is insufficient time to explore the search space. As such, there is a tradeoff between the number of generations used in the MOEA and *N*, the number of repetitions of the RSM to obtain the expected cost of a chromosome. Table 6.4 sought to find the tradeoff value of *N* for DT86. The fields in Table 6.4 are the same as those of Table 6.3. It can be seen from Table 6.4 that $N = 10$ has the lowest average actual multiplicative aggregate of 1.177 and thus can be taken as the tradeoff value for DT86.

Table 6.4 Finding the tradeoff value of N for DT86

N	Expected multiplicative aggregate ($\times 10^6$)	Actual multiplicative aggregate ($\times 10^6$)				Average actual multiplicative aggregate ($\times 10^6$)
		td1	td2	td3	td4	
1	0.897	1.166	1.283	1.089	1.350	1.222
3	1.007	1.136	1.267	1.059	1.279	1.185
5	1.031	1.153	1.259	1.067	1.275	1.189
10	1.038	1.113	1.261	1.061	1.272	1.177
30	1.229	1.272	1.431	1.223	1.431	1.339
50	1.493	1.537	1.705	1.490	1.715	1.612
70	1.796	1.826	1.981	1.754	2.000	1.890

The tradeoff value of N for DT86 has been found to be 10 but this value is problem specific and depends on problem parameters such as the number of customers and the stochastic level of their demands. In order to show how these problem parameters affect the tradeoff value of N, six instances of the VRPSD were created from DT86. The six new test problems differ in the number of customers and the size of the variances of the customer demand distributions. Table 6.5 lists the six test problems. As shown in Table 6.5, the test problems are divided into three groups with 10, 30, and 75 customers. The customer set of the 30-customer test problem is a subset of the set for DT86, while the customer set of the 10-customer test problem is a subset of the set for the 30-customer test problem. Customers are selected such that the size of the maps remains 70×80. Within each group, the test problems are further divided into two sub-groups, one with all customer distributions having a low

variance of 0.002 and the other with a high variance of 76.070. These variance values are the smallest and largest variances of the customer demand distributions of DT86.

Table 6.5 Test problems adapted from DT86

Test problem	Number of customers	Variance of customer demand distributions
vLc10	10	0.002
vHc10		76.070
vLc30	30	0.002
vHc30		76.070
vLc75	75	0.002
vHc75		76.070

The average actual multiplicative aggregates for GEG using the seven *N* settings for the six test problems were found and shown in Table 6.6. As most of the median values of the increase in travel distance and the increase in driver remuneration after implementation using the test demand sets for the low variance problems are zero, the mean values are considered instead for vLc10, vLc30, and vLc75. The lowest average actual multiplicative aggregate for each of the test problems is shown in bold and the corresponding *N* value is the tradeoff value. By comparing the *N* tradeoff values of vLc10 with vHc10, vLc30 with vHc30, and vLc75 with vHc75, it is obvious that given two same-sized problems, i.e. they involve the same number of customers, the decision would be to use a larger value of *N* for the problem where the customers' demands are highly unpredictable, i.e. the variances of the demand distributions are

high, to obtain robust solutions. Similarly, by comparing the N tradeoff values of vLc10 with vLc30 or vLc75, and vHc10 with vHc30 or vHc75, it can be concluded that given two problems with equal stochastic level of customer demands, the decision would be to use a smaller value of N for the problem with a larger search space, i.e. the problem which involves more customers, to allow a more extensive exploration of the search space since a smaller N value corresponds to a larger number of generations used by the MOEA.

Table 6.6 Finding the tradeoff values of N for test problems adapted from DT86

N	Average actual multiplicative aggregate ($\times 10^6$)					
	vLc10	vHc10	vLc30	vHc30	vLc75	vHc75
DET	0.0560		0.1927		0.8649	
1	0.0560	0.1392	0.1922	0.6603	**0.7859**	2.3425
3	0.0545	0.1303	**0.1916**	0.6046	0.8463	2.2450
5	**0.0542**	0.1269	0.2008	**0.6001**	0.8442	**2.1756**
10	0.0581	0.1280	0.2096	0.6108	0.9174	2.2353
30	0.0588	0.1224	0.2354	0.6498	1.1241	2.5852
50	0.0568	0.1237	0.2558	0.6805	1.3497	2.9273
70	0.0609	**0.1216**	0.2754	0.6985	1.5122	3.2273

When an instance of the VRPSD involves customer demand distributions with very low variances, the problem will approach the deterministic VRP which, intuitively, deterministic methods should be sufficient to handle. As such, it would be interesting to compare the performance of DET as described in Section 6.4.3, which

uses the mean demand set to evaluate the fitness of chromosomes, with that of GEG with *N* set to 1, which was concluded in the previous paragraph to be effective in handling test problems with small variances. The performance of DET on the three test problems with low variance is also shown in Table 6.6. From Table 6.6, it can be observed that for vLc10, the performances of the two settings are comparable but as the problem size increases in vLc30, the performance of DET lags behind that of GEG with *N* set to 1. The performance of DET lags further behind for vLc75. From these observations, it can be concluded that deterministic approaches to the VRPSD are sufficient to handle instances with very low stochastic levels and small number of customers but as the problem size gets bigger, their performances deteriorate. This study again shows that for practical problems where the number of customers is more than 10, even though the variances of the customer demand distributions are very low, the stochastic nature of the VRPSD cannot be neglected and it is necessary to design stochastic procedures to handle the stochastic problem.

6.4.5 Choice of *M*

Having analyzed how the value of *N* affects the performance of GEG, this section attempts to study if the parameter *M*, the number of generations of the MOEA before the *N* demand sets used by the RSM are refreshed, has any effect on the performance of GEM. It is to be noted that if $M = 1$, GEM becomes GEG since the *N* demand sets used by the RSM will be refreshed every generation of the MOEA, and if *M* is equal to the maximum number of generations of the MOEA, i.e. the *N* demand sets remain

unchanged throughout the algorithm, and $N = 1$, then GEM becomes DET except that the mean demand set is replaced by a demand set that is randomly generated from the demand distributions of the customers.

Ten simulation runs of three settings with M set to 100, 200, and 300, respectively, were performed on DT86. N was set to 10 for all the three settings. The convergence traces and box plots for the three settings are plotted in Fig. 6.17(a)-(b) and Fig. 6.18(a)-(b), respectively. The plots for GEG ($M = 1$), $M = 10$ (the GEM setting in Section 6.4.3), and DET are also plotted for comparison. From the figures, it can be seen that the performance of GEM is the same regardless of the value of M. The average travel distance and the average driver remuneration of the Pareto solutions found are almost the same. The robustness of the solutions is also comparable. The large disparity between $M = 300$ and DET in terms of how well the expected costs of solutions estimate the actual costs also highlights the contribution of the RSM to finding robust solutions to the VRPSD since the major difference between the two settings is the RSM's ability to use N randomly generated demand sets to evaluate the fitness of chromosomes.

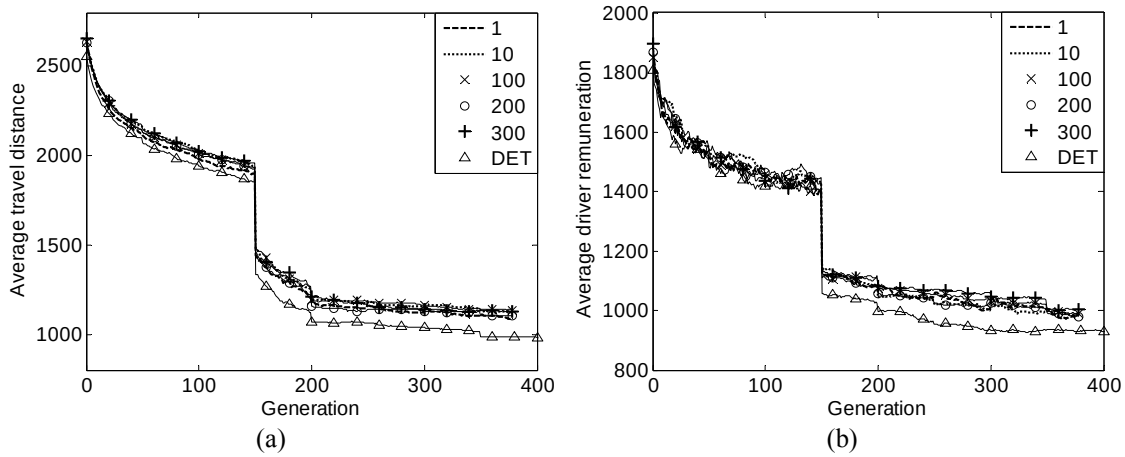


Fig. 6.17 (a) Average travel distance and (b) average driver remuneration of non-dominated solutions of GEM using different *M* values

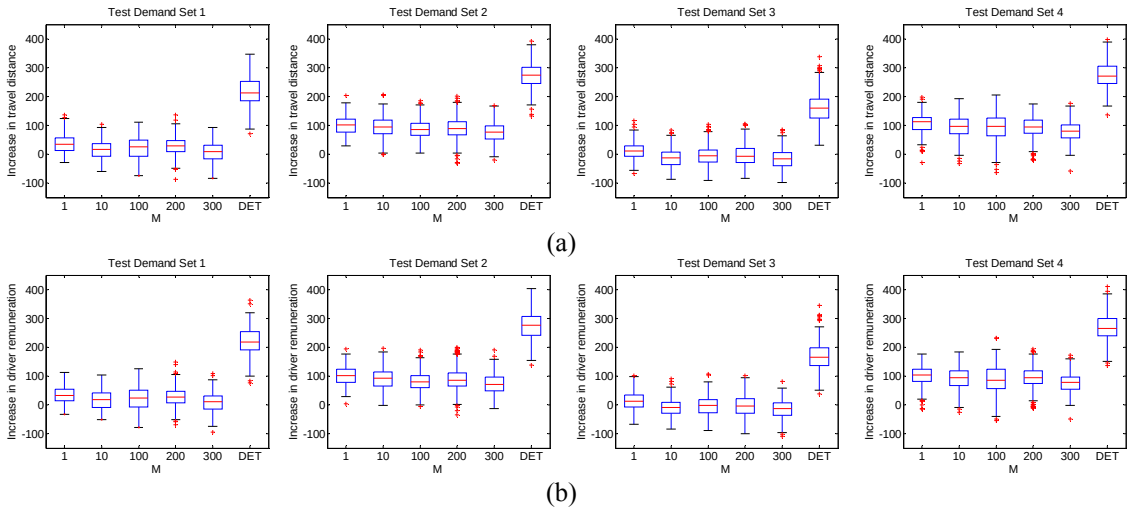


Fig. 6.18 Increase in (a) travel distance and (b) driver remuneration after implementing Pareto solutions of GEM using different *M* values

6.4.6 Performance of MOEA on Other Test Problems

As mentioned at the beginning of Section 6.4, it was highlighted by Bianchi *et al.* (2004) that there is no commonly used benchmark for the VRPSD in the literature. As such, most of the performance analysis of the MOEA up till now has been

conducted on DT86. It would be important that the performance of the MOEA is not compromised over a wide range of problems. Therefore, it is the goal of this section to show that the performance of the MOEA, as shown in the earlier results, is reproducible on a few test problems adapted from the well-established VRPTW benchmark problems designed by Solomon (1987).

The original design of Solomon's VRPTW benchmark problems (Solomon, 1987) highlights several factors, which can affect the behavior of routing and scheduling algorithms, of which the topology of customers is the concern of this section. The benchmark problems consist of 56 data sets which can be categorized into three main classes based on the spatial distribution of customers. For the Type-R problems, all the customers are remotely located, whereas for the Type-C problems, the customers are grouped into a few clusters. The Type-RC problems consist of a mixture of remote and clustered customers. Customer information, such as the number of customers (each instance consists of 100 customers), their locations, demands, and service times, within each category is identical. Using these three sets of customer information, three test problems are created. Like Dror and Trudeau (1986), to adapt the customer information to the VRPSD, the original demand quantity is used as the mean demand of each customer. The standard deviation of the demand distribution of each customer is generated using a uniform random number generator so that it falls between zero and one-third of the mean demand of the customer. The stochastic versions of the Type-R, Type-C, and Type-RC problems of Solomon (1987) will be referred to as Type-RS, Type-CS, and Type-RCS, respectively.

In order to show that the effectiveness of the local search operators and the ability of the RSM to produce robust solutions are reproducible on Type-RS, Type-CS, and Type-RCS, the performance of the MOEA is compared with two other settings. The settings are described in Table 6.7.

Table 6.7 Description of settings for performance testing

Setting	Local search	Fitness evaluation
MOEA	RAN	RSM (GEG with $N = 10$)
NLSRSM	NLS	RSM (GEG with $N = 10$)
LSDET	RAN	DET

Ten simulation runs of each of the three settings were conducted on Type-RS, Type-CS, and Type-RCS. The results are tabulated in Table 6.8. The fields in Table 6.8 are the same as those of Table 6.3, except for increase in multiplicative aggregate, which is the difference between average actual multiplicative aggregate and expected multiplicative aggregate. It is used as a measure of the robustness of the solutions, with a smaller value indicating solutions having expected costs that are good estimates of the actual costs of implementation on the randomly generated test demand sets.

Table 6.8 Performance of MOEA on Type-RS, Type-CS, and Type-RCS

Test problem	Setting	Expected multiplicative aggregate ($\times 10^6$)	Actual multiplicative aggregate ($\times 10^6$)				Average actual multiplicative aggregate ($\times 10^6$)	Increase in multiplicative aggregate ($\times 10^6$)
			td1	td2	td3	td4		
Type-RS	MOEA	1.432	1.598	1.650	1.512	1.537	1.574	0.142
	NLSRSM	4.015	4.262	4.352	4.133	4.138	4.222	0.207
	LSDET	1.103	1.622	1.605	1.461	1.477	1.541	0.438
Type-CS	MOEA	1.027	0.942	1.095	1.223	0.833	1.023	-0.004
	NLSRSM	3.933	3.782	4.023	4.378	3.726	3.978	0.045
	LSDET	0.658	1.061	1.229	1.344	0.962	1.149	0.491
Type-RCS	MOEA	1.433	1.855	1.754	1.747	1.868	1.806	0.373
	NLSRSM	4.902	5.593	5.332	5.515	5.556	5.499	0.597
	LSDET	0.991	2.080	1.967	1.878	2.206	2.032	1.041

Comparing the average actual multiplicative aggregates for the three settings on the three test problems, it can be seen that for Type-CS and Type-RCS, the MOEA performed the best among the three settings but for Type-RS, its performance is marginally poorer than LSDET. One probable explanation could be that the N value of 10 used in the simulations on Type-RS is not the tradeoff value. However, the fact that the MOEA performed favorably on Type-CS and Type-RCS could also imply that the tradeoff value of N for a particular problem, on top of being dependent on problem parameters such as the number of customers and the stochastic level of their demands, may also be affected by the spatial distribution of the customers. Putting this aside, comparing the increases in multiplicative aggregates, it is obvious that for

all the three problems, the solutions produced by the MOEA are more robust than those obtained by the other two settings.

Having discussed the performance of the MOEA on the three test problems, the effectiveness of the local search operators and the ability of the RSM to produce robust solutions are also evident from Table 6.8. It can be seen from Table 6.8 that the settings which utilize local search, namely the MOEA and LSDET, achieved lower average actual multiplicative aggregates than NLSRSM. Comparing the increases in multiplicative aggregates, it can also be seen from Table 6.8 that the settings which utilize the RSM, namely the MOEA and NLSRSM, were able to produce solutions that are more robust than those obtained by LSDET. These observations highlight that the effectiveness of the local search operators and the ability of the RSM to produce robust solutions as achieved earlier on DT86 are reproducible on the three VRPSD instances adapted from Solomon's VRPTW benchmark problems (Solomon, 1987).

6.4.7 Significance of the RSM

In this chapter, the RSM is implemented by using demand sets randomly generated based on the customers' demand distributions. In the actual implementation of the algorithm, there may not be a need to randomly generate the demand sets if the company keeps past demand records of their customers. These past records can be used to provide the demand sets for the RSM to operate on. This approach can be particularly useful if the customers' demand distributions are not known or if the

customers' demands change for different days of the week. Take for example the delivery of beer, the customers' demands for beer are usually higher on the weekends when people get together for happy hour. As such, instead of having a separate demand distribution for a particular customer for each day of the week, using the customer's past demand records for the appropriate day would allow us to easily come up with demand sets for the operations of the RSM.

6.5 Summary

This chapter studied the routing of vehicles with limited capacity from a central depot to a set of geographically dispersed customers, whose actual demands are revealed only when the vehicles arrive at their locations. The solution to this vehicle routing problem with stochastic demand (VRPSD) involves the optimization of routing schedules with minimum travel distance, driver remuneration, and number of vehicles, subject to a number of constraints such as time windows and vehicle capacity. A multi-objective evolutionary algorithm (MOEA), which incorporates two VRPSD-specific heuristics for local exploitation and a route simulation method to evaluate the fitness of solutions, has been proposed in this chapter to solve the multi-objective optimization problem. A new way of assessing the quality of solutions to the VRPSD on top of comparing their expected costs has also been presented. It has been shown that the algorithm is capable of finding useful tradeoff solutions for the VRPSD and the solutions are robust to the stochastic nature of the problem. The developed algorithm has been further validated on a few VRPSD instances adapted

from Solomon's vehicle routing problem with time windows (VRPTW) benchmark problems.

Chapter 7

Conclusions

Multi-objective evolutionary algorithms (MOEAs) are a class of stochastic optimization techniques that have been proving to be very efficient and effective in solving sophisticated multi-objective optimization problems where conventional optimization tools have been found to be inadequate. MOEAs operate on a population of solutions, which allows them to sample multiple candidate solutions simultaneously and find multiple optimal solutions in a single simulation run. MOEAs are well-known for their ability to solve non-linear and combinatorial problems. They are also often noted for searching large, multi-modal spaces effectively, without requiring any gradient or problem-specific information. These aspects of MOEAs have made them natural solvers for multi-objective scheduling problems, which provided the main motivation for the research documented in this thesis.

7.1 Contributions

The primary aim of this thesis is to present an investigation on the application of MOEAs to solve a few scheduling problems with vastly different characteristics. A generic MOEA framework has been devised in Chapter 3. Problem-specific operators are then designed to adapt the MOEA to solve the different scheduling problems considered in this thesis.

In Chapter 4, the exam timetabling problem (ETTP) has been considered as a multi-objective optimization problem that involves the minimization of the two objectives of number of clashes and number of periods in a timetable. The MOEA, featured with variable-length chromosome representation, graph coloring heuristics, goal-based Pareto ranking scheme, and two local search operators of micro-genetic algorithm and hill-climber, has been presented. The proposed MOEA differs from existing approaches in that it considers timetable length as an objective to be optimized rather than expecting an input from the timetable planner. It generates a Pareto set of solutions within the desired range of timetable lengths instead of producing single-length timetables. It has been demonstrated that such an approach is more general and would still be able to function effectively even without any prior timetable length information. The results have also shown that the MOEA is able to generate shorter clash-free timetables which can never be found by existing approaches. On top of these, the MOEA has also performed well in comparison with seven other recent and established optimization techniques. The MOEA is able to produce the best results for four out of the seven publicly available datasets tested.

In Chapter 5, the berth allocation problem (BAP), which involves the minimization of the three objectives of makespan, waiting time, and degree of deviation from a predetermined priority schedule, has been studied. These objectives are considered with the interests of both port and ship operators in mind. Three primary features, including a local search heuristic, a hybrid solution decoding scheme, and an optimal berth insertion procedure, have been designed to adapt the MOEA to solve the BAP. The proposed MOEA differs from most existing single-objective-based approaches in that it optimizes all objectives concurrently without the need of aggregating them into a compromise function. Given the intricate relationships between the three objectives that have been uncovered in this work, the multi-objective approach appears to be the natural choice for tackling the BAP. It generates a Pareto set of berth schedules from which the port management can select a desirable solution for implementation. In addition, the effects that the three proposed features have on the quality of berth schedules have been studied. It has been shown and validated that the features play a pivotal role in the optimization performance of the MOEA.

A capacity and time constrained vehicle routing problem with stochastic demand (VRPSD) has been considered in Chapter 6. The problem is inherently a multi-objective optimization problem that involves the optimization of routes for multiple vehicles to minimize the three objectives of travel distance, driver remuneration, and number of vehicles required. The MOEA, featured with two VRPSD-specific local search heuristics, has been presented. To evaluate the cost of a VRPSD solution, which is stochastic, a route simulation method (RSM) has also been proposed and

incorporated with the MOEA. Without the need of aggregating the multiple objectives of the VRPSD into a compromise function, the MOEA optimizes all the objectives concurrently, providing advantages such as improved routing solutions and the exploration of a larger search space. The effectiveness of the two VRPSD-specific local search heuristics and the various settings in which local exploitation is incorporated with the MOEA have been studied. A new way of assessing the quality of solutions to the VRPSD on top of comparing their expected costs has also been proposed. Extensive simulations have been performed to show that the solutions obtained by the MOEA, equipped with the RSM, are robust to the stochastic nature of the problem. The expected costs of such solutions are good approximations of the actual costs of implementing the solutions, thus providing the logistic manager with accurate information based on which decision will be made.

7.2 Future Works

Although this thesis has provided a detailed study of the application of MOEAs to solve multi-objective scheduling problems, there is much room for expansion in future works. One direction of investigation pertaining to the BAP involves the design of an encompassing multi-objective optimization framework that is capable of solving the BAP and the quay crane assignment problem simultaneously. In the current BAP model, it is assumed that the time required to load or unload containers for each ship is known and fixed. However, it is intuitive that the time for container handling depends on the number of quay cranes, which are responsible for moving

containers between a ship and a berth, assigned to the ship. Although existing methods approach these two problems independently, the two problems are very closely related in that the solution of one directly affects the other problem. A co-evolutionary algorithm could be developed to exploit the interaction between the two problems for better optimization performance.

The framework suggested above could also be applied to solve the ETTP with the course timetabling problem (CTTP). The CTTP involves scheduling university courses into a weekly timetable. Since the two problems share the same set of enrolment data, the solution obtained for either one of the problems could be exploited to assist in solving the other problem. Another research direction concerning the ETTP involves the problem of assigning exams to rooms. The work in Chapter 4 has focused on the temporal aspect of the ETTP, i.e. the allocation of exams to periods. It has to be acknowledged that for a more complete treatment of the timetabling problem, the spatial aspect of the problem, i.e. the assignment of exams to rooms, has to be considered as well. This opens up another dimension of the multi-objective optimization problem.

The performance validation of the MOEA proposed for solving the VRPSD in Section 6.4.6 also opens up the prospects for future work. In that section, the MOEA was tested on three test problems adapted from the vehicle routing problem with time windows (VRPTW) benchmark problems of Solomon (1987). More test problems can actually be created by varying some of the problem parameters such as the geographical location of the depot, the customer to vehicle ratio, and the stochastic level of the customer demands. Larger test problems can also be adapted from the

extended Solomon VRPTW benchmark problems of Homberger and Gehring (1999). This would create a complete set of benchmark problems for the VRPSD, which can be used for further simulation studies to understand more behaviors of the problem and also acts as a basis for comparison of algorithm performance, which has been lacking hitherto. It can also be seen from the studies in Section 6.4.4 that the performance of the MOEA for solving the VRPSD is affected by how close the value of N is to the tradeoff value of the problem. It would be computationally intensive to find the tradeoff value of N for a given problem to ensure that the performance of the MOEA is not compromised. Therefore, it would be useful if a rule of thumb can be developed such that it would be possible to select the value of N to be set in the MOEA by inspecting just a few parameters of the problem.

Bibliography

S. Abdullah, S. Ahmadi, E. K. Burke, and M. Dror, "Investigating Ahuja-Orlin's large neighbourhood search approach for examination timetabling", *OR Spectrum*, vol. 29, no. 2, pp. 351 – 372, 2007a.

S. Abdullah, S. Ahmadi, E. K. Burke, M. Dror, and B. McCollum, "A tabu-based large neighbourhood search methodology for the capacitated examination timetabling problem", *Journal of the Operational Research Society*, vol. 58, pp. 1494 – 1502, 2007b.

S. Ahmadi, R. Barrone, P. Cheng, P. Cowling, and B. McCollum, "Perturbation based variable neighbourhood search in heuristic space for examination timetabling problem", in *Proceedings of Multidisciplinary International Scheduling: Theory and Applications*, *MISTA 2003*, Nottingham, UK, pp. 155 – 171, 2003.

R. K. Ahuja, J. B. Orlin, and D. Sharma, "Multiexchange neighbourhood search algorithm for capacitated minimum spanning tree problem", *Mathematical Programming*, vol. 91, pp. 71 – 97, 2001.

U. Aickelin and K. A. Dowsland, "An indirect genetic algorithm for a nurse scheduling problem", *Computers & Operations Research*, vol. 31, no. 5, pp. 761 – 778, 2004.

H. Asmuni, E. K. Burke, J. M. Garibaldi, and B. McCollum, "Fuzzy multiple heuristic orderings for examination timetabling", in *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2004*, Pittsburg, PA, USA, E. K. Burke and M. Trick (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 3616, pp. 334 – 353, 2005.

H. Asmuni, E. K. Burke, J. M. Garibaldi, and B. McCollum, "A novel fuzzy approach to evaluate the quality of examination timetabling", in *Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2006*, Brno, Czech Republic, E. K. Burke and H. Rudová (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 3867, pp. 327 – 346, 2007.

N. Balakrishnan, A. Lucena, and R. T. Wong, "Scheduling examinations to reduce second order conflicts", *Computers and Operations Research*, vol. 19, pp. 353 – 361, 1992.

V. A. Bardadym, "Computer-aided school and university timetabling: The new wave", in *Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 1995*, Edinburgh, Scotland, E. K. Burke and P. Ross (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1153, pp. 22 – 45, 1996.

M. Basseur, F. Seynhaeve, and E. Talbi, "Design of multi-objective evolutionary algorithms: application to the flow-shop scheduling problem", in *Proceedings of the 2002 Congress on Evolutionary Computation*, *CEC 2002*, Honolulu, HI, USA, vol. 2, pp. 1151 – 1156, 2002.

D. J. Bertsimas, P. Jaillet, and A. R. Odoni, "A priori optimization", *Operations Research*, vol. 38, pp. 1019 – 1033, 1990.

D. J. Bertsimas, P. Chervi, and M. Peterson, "Computational approaches to stochastic vehicle routing problems", *Transportation Science*, vol. 29, no. 4, pp. 342 – 352, 1995.

L. Bianchi, M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto, "Metaheuristics for the vehicle routing problem with stochastic demands", in *Proceedings of the 8th International Conference on Parallel Problem Solving from Nature*, Birmingham, UK, vol. 3242 of Lecture Notes in Computer Science, pp. 450 – 460, 2004.

L. Bianchi, M. Birattari, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto, "Hybrid metaheuristics for the vehicle routing problem with stochastic demands", *Journal of Mathematical Modelling and Algorithms*, vol. 5, no. 1, pp. 91 – 110, 2006.

B. Bilgin, E. Özcan, and E. E. Korkmaz, "An experimental study on hyper-heuristics and exam timetabling", in *Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2006*, Brno, Czech Republic, E. K. Burke and H. Rudová (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 3867, pp. 394 – 412, 2007.

P. Bosman and D. Thierens, "The balance between proximity and diversity in multi-objective evolutionary algorithms", *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 174 – 188, 2003.

S. C. Brailsford, C. N. Potts, and B. M. Smith, "Constraint satisfaction problems: algorithms and applications", *European Journal of Operational Research*, vol. 119, pp. 557 – 581, 1999.

D. Brelaz, "New methods to color the vertices of a graph", *Communication of the ACM*, vol. 22, no. 4, pp. 251 – 256, 1979.

S. Broder, "Final examination scheduling", *Communications of the ACM*, vol. 7, pp. 494 – 498, 1964.

G. G. Brown, S. Lawphongpanich, and K. P. Thurman, "Optimizing ship berthing", *Naval Research Logistics*, vol. 41, pp. 1 – 15, 1994.

G. G. Brown, K. J. Cormican, S. Lawphongpanich, and D. B. Widddis, "Optimizing submarine berthing with a persistence incentive", *Navel Research Logistics*, vol. 44, pp. 301 – 318, 1997.

B. Bullnheimer, "An examination scheduling model to maximize students study time", in *Proceedings of the 2nd International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 1997*, Toronto, Canada, E. K. Burke and M. W. Carter (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1408, pp. 78 – 91, 1998.

R. L. Burdett and E. Kozan, "Evolutionary algorithms for flowshop sequencing with non-unique jobs", *International Transactions in Operational Research*, vol. 7, no. 4, pp. 401 – 418, 2000.

E. K. Burke, D. G. Elliman, P. H. Ford, and R. F. Weare, "Specialised recombinative operators for the timetabling problem", in *Evolutionary Computing: AISB Workshop*, Sheffield, UK, T. Fogarty (Ed.), Springer-Verlag Lecture Notes in Computer Science, vol. 993, pp. 75 – 85, 1995.

E. K. Burke, J. P. Newall, and R. F. Weare, "A memetic algorithm for university exam timetabling", in *Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 1995*, Edinburgh, Scotland, E. K. Burke and P. Ross (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1153, pp. 241 – 250, 1996a.

E. K. Burke, D. G. Elliman, P. H. Ford, and R. F. Weare, "Examination timetabling in british universities - a survey", in *Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 1995*, Edinburgh, Scotland, E. K. Burke and P. Ross (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1153, pp. 76 – 90, 1996b.

E. K. Burke, K. Jackson, J. H. Kingston, and R. Weare, "Automated university timetabling: the state of the art", *The Computer Journal*, vol. 40, no. 9, pp. 565 – 571, 1997.

E. K. Burke, J. P. Newall, and R. F. Weare, "A simple heuristically guided search for the timetable problem", in *Proceedings of the International ICSC Symposium on*

Engineering of Intelligent Systems, *EIS 1998*, Spain, E. Alpaydin and C. Fyte (Eds.), pp. 574 – 579, 1998a.

E. K. Burke, J. P. Newall, and R. F. Weare, "Initialization strategies and diversity in evolutionary timetabling", *Evolutionary Computation Journal*, vol. 6, no. 1, pp. 81 – 103, 1998b.

E. K. Burke and J. P. Newall, "A multistage evolutionary algorithm for the timetable problem", *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 1, pp. 63 – 74, 1999.

E. K. Burke, Y. Bykov, and S. Petrovic, "A multicriteria approach to examination timetabling", in *Proceedings of the 3rd International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2000*, Konstanz, Germany, E. K. Burke and W. Erben (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2079, pp. 118 – 131, 2001.

E. K. Burke and S. Petrovic, "Recent research directions in automated timetabling", *European Journal of Operational Research*, vol. 140, no. 2, pp. 266 – 280, 2002.

E. K. Burke, S. Petrovic, and R. Qu, "Case-based heuristic selection for examination timetabling", in *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning*, *SEAL 2002*, Singapore, pp. 277 – 281, 2002.

E. K. Burke and J. P. Newall, "Enhancing timetable solutions with local search methods", in *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2002*, Gent, Belgium, E. K. Burke and P. De Causmaecker (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2740, pp. 195 – 206, 2003.

E. K. Burke and J. D. Landa Silva, "The design of memetic algorithms for scheduling and timetabling problems", *Recent Advances in Memetic Algorithms and Related Search Technologies*, *Studies in Fuzziness and Soft Computing*, W. E. Hart, N. Krasnogor, and J. E. Smith (Eds.), Springer, vol. 166, pp. 289 – 312, 2004.

E. K. Burke and J. P. Newall, "Solving examination timetabling problems through adaptation of heuristic orderings", *Annals of Operational Research*, vol. 129, pp. 107 – 134, 2004.

E. K. Burke, J. Kingston, and D. de Werra, "Applications to timetabling", in *Handbook of Graph Theory*, J. Gross and J. Yellen (Eds.), Chapman Hall/CRC Press, pp. 445 – 474, 2004a.

E. K. Burke, Y. Bykov, J. P. Newall, and S. Petrovic, "A time-predefined local search approach to exam timetabling problems", *IIE Transactions*, vol. 36, no. 6, pp. 509 – 528, 2004b.

E. K. Burke, M. Dror, S. Petrovic, and R. Qu., "Hybrid graph heuristics in hyper-heuristics applied to exam timetabling problems", *The Next Wave in Computing, Optimization, and Decision Technologies*, B. L. Golden, S. Raghavan, and E. A. Wasil (Eds.), Springer, pp. 79 – 91, 2005.

E. K. Burke, A. J. Eckersley, B. McCollum, S. Petrovic, and R. Qu, "Hybrid variable neighbourhood approaches to university exam timetabling", School of CSiT, University of Nottingham, Technical Report NOTTCS-TR-2006-2, 2006a.

E. K. Burke, S. Petrovic, and R. Qu, "Case-based heuristic selection for timetabling problems", *Journal of Scheduling*, vol. 9, pp. 115 – 132, 2006b.

E. K. Burke, B. McCollum, A. Meisels, S. Petrovic, and R. Qu, "A graph-based hyper-heuristic for educational timetabling problems", *European Journal of Operational Research*, vol. 176, pp. 177 – 192, 2007.

M. Caramia, P. Dell'Olmo, and G. F. Italiano, "New algorithms for examination timetabling", in *Algorithm Engineering 4th International Workshop, WAE 2000*, Saarbrücken, Germany, S. Näher and D. Wagner (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1982, pp. 230 – 241, 2001.

J. Carretero, F. Xhafa, and A. Abraham, "Genetic algorithm based schedulers for grid computing systems", *International Journal of Innovative Computing, Information and Control*, vol. 3, no. 5, pp. 1053 – 1071, 2007.

M. W. Carter, "A survey of practical applications of examination timetabling algorithms", *Operations Research*, vol. 34, no. 2, pp. 193 – 202, 1986.

M. W. Carter and G. Laporte, "Recent developments in practical examination timetabling", in *Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 1995*, Edinburgh, Scotland, E. K. Burke and P. Ross (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1153, pp. 3 – 21, 1996.

M. W. Carter, G. Laporte, and S. Y. Lee, "Examination timetabling: algorithmic strategies and applications", *The Journal of the Operational Research Society*, vol. 47, no. 3, pp. 373 – 383, 1996

M. W. Carter and D. G. Johnson, "Extended partition initialization in examination timetabling", *The Journal of the Operational Research Society*, vol. 52, pp. 538 – 544, 2001.

S. Casey and J. Thompson, "GRASPing the examination scheduling problem", in *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2002*, Gent, Belgium, E. K. Burke and P. De

Causmaecker (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2740, pp. 232 – 244, 2003.

C. K. Chan, H. B. Gooi, and M. H. Lim, "Co-evolutionary algorithm approach to a university timetable system", in *Proceedings of the 2002 Congress on Evolutionary Computation*, *CEC 2002*, Honolulu, HI, USA, vol. 2, pp. 1946 – 1951, 2002.

A. Charnes and W. Cooper, "Chance-constrained programming", *Management Science*, vol. 6, pp. 73 – 79, 1959.

A. Charnes and W. Cooper, "Deterministic equivalents for optimizing and satisfying under chance constraints", *Operations Research*, vol. 11, pp. 18 – 39, 1963.

C. H. Chen, S. D. Wu, and L. Dai, "Algorithm comparison for manufacturing scheduling problems", in *Proceedings of the 36th IEEE Conference on Decision and Control*, San Diego, CA, USA, 1997, vol. 2, pp. 1214 – 1215, 1997.

C. A. Coello Coello and G. T. Pulido, "Multiobjective optimization using a micro-genetic algorithm", in *Proceedings of the 2001 Genetic and Evolutionary Computation Conference, GECCO 2001*, San Francisco, CA, USA, L. Spector, E. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. K. Burke (Eds.), Morgan Kaufmann Publishers, pp. 274 – 282, 2001.

D. A. Coley, *An Introduction to Genetic Algorithms for Scientists and Engineers*, World Scientific Publishing, 1999.

D. W. Corne, J. D. Knowles, and M. J. Oates, "The Pareto envelope-based selection algorithm for multiobjective optimization", in *Proceedings of the 6th International Conference on Parallel Problem Solving from Nature*, *PPSN VI*, Paris, France, M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, and H. P. Schwefel (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1917, pp. 839 – 848, 2000.

P. Côté, T. Wong, and R. Sabouri, "Application of a hybrid multi-objective evolutionary algorithm to the uncapacitated exam proximity problem", in *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2004*, Pittsburg, PA, USA, E. K. Burke and M. Trick (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 3616, pp. 151 – 168, 2005.

P. Cowling, G. Kendall, and L. Han, "An investigation of a hyperheuristic genetic algorithm applied to a trainer scheduling problem", in *Proceedings of the 2002 Congress on Evolutionary Computation*, *CEC 2002*, Honolulu, HI, USA, vol. 2, pp. 1185 – 1190, 2002.

P. Dasgupta, P. P. Chakrabarti, and S. C. Desarkar, *Multiobjective Heuristic Search: An Introduction to Intelligent Search Methods for Multicriteria Optimization*, Springer-Verlag, New York, 1999.

P. David, "A constraint-based approach for examination timetabling using local repair techniques", in *Proceedings of the 2nd International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 1997*, Toronto, Canada, E. K. Burke and M. W. Carter (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1408, pp. 169 – 186, 1998.

R. Dawkins, *The Selfish Gene*, Oxford University Press, New York, 1976.

D. de Werra, "An introduction to timetabling", *European Journal of Operational Research*, vol. 19, pp. 151 – 162, 1985.

K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, John Wiley & Sons, New York, 2001.

K. Deb and H.-G. Beyer, "Self-adaptive genetic algorithms with simulated binary crossover", *Evolutionary Computation Journal*, vol. 9, no. 2, pp. 197 – 221, 2001.

K. Deb, A. Anand, and D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter optimization", *Evolutionary Computation Journal*, vol. 10, no. 4, pp. 371 – 395, 2002a.

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II", *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182 – 197, 2002b.

K. Deb, M. Mohan, and S. Mishra, "Evaluating the ε-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions", *Evolutionary Computation Journal*, vol. 13, no. 4, pp. 501 – 525, 2005.

K. Deb and S. Tiwari, "Omni-optimizer: A generic evolutionary algorithm for single and multi-objective optimization", *European Journal of Operational Research*, vol. 185, no. 3, pp. 1062 – 1087, 2008.

L. Di Gaspero and A. Schaerf, "Tabu search techniques for examination timetabling", in *Proceedings of the 3rd International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2000*, Konstanz, Germany, E. K. Burke and W. Erben (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2079, pp. 104 – 117, 2001.

K. A. Dowsland, "Simulated annealing solutions for multi-objective scheduling and timetabling", *Modern Heuristic Search Methods*, V. J. R. Smith, I. H. Osman, C. R. Reeves, and G. D. Smith (Eds.), John Wiley, pp. 155 – 166, 1996.

K. A. Dowsland and J. Thompson, "Ant colony optimization for the examination scheduling problem", *The Journal of Operational Research Society*, vol. 56, pp. 426 – 438, 2005.

G. Dozier, J. Brown, and D. Bahler, "Solving small and large scale constraint satisfaction problems using a heuristic-based microgenetic algorithm", in *Proceedings of the 1st IEEE International Conference on Evolutionary Computation*, *CEC 1994*, Piscataway, NJ, USA, vol. 1, pp. 306 – 311, 1994.

M. Dror, M. O. Ball, and B. L. Golden, "Computational comparison of algorithms for inventory routing", *Annals of Operations Research*, vol. 4, pp. 3 – 23, 1985.

M. Dror and P. Trudeau, "Stochastic vehicle routing with modified savings algorithm", *European Journal of Operational Research*, vol. 23, no. 2, pp. 228 – 235, 1986.

M. Dror, "Modeling vehicle routing with uncertain demands as a stochastic program: Properties of the corresponding solution", *European Journal of Operational Research*, vol. 64, no. 3, pp. 432 – 441, 1993.

T. A. Duong and K. H. Lam, "Combining constraint programming and simulated annealing on university exam timetabling", in *Proceedings of the 2nd International Conference in Computer Sciences*, *Research*, *Innovation & Vision for the Future*, *RIVF 2004*, Hanoi, Vietnam, pp. 205 – 210, 2004.

M. Eley, "Ant algorithms for the exam timetabling problem", in *Proceedings of the 6th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2006*, Brno, Czech Republic, E. K. Burke and H. Rudová (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 3867, pp. 364 – 382, 2007.

W. Erben, "A grouping genetic algorithm for graph colouring and exam timetabling", in *Proceedings of the 3rd International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2000*, Konstanz, Germany, E. K. Burke and W. Erben (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2079, pp. 132 – 156, 2001.

W. Erben and P. Y. Song, "A hybrid grouping genetic algorithm for examination timetabling", in *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2004*, Pittsburg, PA, USA, E. K. Burke and M. Trick (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 3616, pp. 487 – 490, 2005.

H. Eskandari, C. D. Geiger, and G. B. Lamont, "FastPGA: A dynamic population sizing approach for solving expensive multiobjective optimization problems", in *Proceedings of the 4th International Conference on Evolutionary Multi-Criterion Optimization*, *EMO 2007*, Matsushima, Japan, Hutchison *et al*. (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 4403, pp. 141 – 155, 2007.

L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*, John Wiley, New York, 1966.

C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization", in *Proceedings of the 5th International Conference on Genetic Algorithms*, San Mateo, USA, pp. 416 – 423, 1993.

C. M. Fonseca and P. J. Fleming, "Multiobjective optimization and multiple constraint handling with evolutionary algorithms - Part I: A unified formulation", *IEEE Transactions on Systems, Man and Cybernetics, Part A*, vol. 28, no. 1, pp. 26 – 37, 1998.

H. M. Foo, "The application of genetic algorithms to the berth allocation problem", Dept. of Computer Science, Faculty of Science, National University of Singapore, Singapore, Master's Thesis, 2000.

T. A. Gani, A. T. Khader, and R. Budiarto, "Optimizing examination timetabling using a hybrid evolution strategies", in *Proceedings of the 2nd International Conference on Autonomous Robots and Agents*, Palmerston North, New Zealand, pp. 345 – 349, 2004.

M. R. Garey and D. S. Johnson, *Computers and Intractablility: A Guide to the Theory of NP-completeness*, W. H. Freeman, San Francisco, 1979.

M. Gendreau, A. Hertz, and G. Laporte, "A tabu search heuristic for the vehicle routing problem", *Management Scie*nce, vol. 40, pp. 1276 – 1290, 1994.

M. Gendreau, G. Laporte, and R. Séguin, "An exact algorithm for the vehicle routing problem with stochastic demands and customers", *Transportation Science*, vol. 29, pp. 143 – 155, 1995.

M. Gendreau, G. Laporte, and R. Séguin, "Stochastic vehicle routing", *European Journal of Operational Research*, vol. 88, no. 1, pp. 3 – 12, 1996a.

M. Gendreau, G. Laporte, and R. Séguin, "A tabu search heuristic for the vehicle routing problem with stochastic demands and customers", *Operations Research*, vol. 44, no. 3, pp. 469 – 477, 1996b.

K. S. Goh, A. Lim, and B. Rodrigues, "Sexual selection for genetic algorithms", *Artificial Intelligence Review*, vol. 19, no. 2, pp. 123 – 152, 2003.

A. Goicoechea, D. R. Hansen, and L. Duckstein, *Multiobjective Decision Analysis with Engineering and Business Applications*, John Wiley & Sons, New York, 1982.

D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.

Y. Guan, W.-Q. Xiao, R. K. Cheung, and C.-L. Li, "A multiprocessor task scheduling model for berth allocation: heuristic and worst case analysis", *Operations Research Letters*, vol. 30, pp. 343 – 350, 2002.

P. Hansen and N. Mladenovic, "Variable neighbourhood search: principles and applications", *European Journal of Operational Research*, vol. 130, pp. 449 – 467, 2001.

E. Hart, P. M. Ross, and J. Nelson, "Scheduling chicken catching - An investigation into the success of a genetic algorithm on a real world scheduling problem", *Annals of Operations Research*, vol. 92, pp. 363 – 380, 1999.

P. V. Hentenryck, "Constraint satisfaction in logic programming", *Logic Programming Series*, MIT Press, Cambridge, MA, 1989.

K. S. Hindi, H. Yang, and K. Fleszar, "An evolutionary algorithm for resource-constrained project scheduling", *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 512 – 518, 2002.

J. Holland, *Adaptation in Natural and Artificial Systems*, The MIT Press, 1975.

J. Homberger and H. Gehring, "Two evolutionary meta-heuristics for the vehicle routing problem with time windows", *INFORMS Journal on Computing*, vol. 37, no. 3, pp. 297 – 318, 1999.

J. Horn and N. Nafpliotis, "Multiobjective optimization using the niched Pareto genetic algorithm", Illinois Genetic Algorithms Laboraatory, IlliGAL, University of Illinois, Report no. 930005, 1993.

J. Horn, N. Nafpliotis, and D. E. Goldberg, "A niched Pareto genetic algorithm for multiobjective optimization", in *Proceedings of the 1st IEEE International Conference on Evolutionary Computation*, *CEC 1994*, Piscataway, NJ, USA, vol. 1, pp. 82 – 87, 1994.

X.-B. Hu and E. Di Paolo, "An efficient genetic algorithm with uniform crossover for the multi-objective airport gate assignment problem", *Multi-Objective Memetic*

Algorithms, K. C. Tan, C. K. Goh, and Y. S. Ong (Eds.), Springer-Verlag, pp. 71 – 90, 2009.

M. Hussain, B. Kimiaghalam, A. Ahmedzadeh, and A. Homaifar, "Multi-robot scheduling using evolutionary algorithms", in *Proceedings of the 5th Biannual World Automation Congress 2002*, vol. 13, pp. 233 – 238, 2002.

N. Hussin, "Tabu search based hyper-heuristic approaches for examination timetabling", PhD Thesis, Department of Computer Science, University of Nottingham, UK, 2005.

H. S. Hwang, "An improved model for vehicle routing problem with time constraint based on genetic algorithm", *Computers & Industrial Engineering*, vol. 42, pp. 361 – 369, 2002.

A. Imai, K. Nagaiwa, and W. T. Chan, "Efficient planning of berth allocation for container terminals in Asia", *Journal of Advanced Transportation*, vol. 31, pp. 75 – 94, 1997.

A. Imai, E. Nishimura, and S. Papadimitriou, "The dynamic berth allocation problem for a container port", *Transportation Research Part B: Methodological*, vol. 35, no. 4, pp. 401 – 417, 2001.

A. Imai, E. Nishimura, and S. Papadimitriou, "Berth allocation with service priority", *Transportation Research Part B: Methodological*, vol. 37, no. 5, pp. 437 – 457, 2003.

A. Imai, X. Sun, E. Nishimura, and S. Papadimitriou, "Berth allocation in a container port: using a continuous location space approach", *Transportation Research Part B: Methodological*, vol. 39, no. 3, pp. 199 – 221, 2005.

A. Imai, E. Nishimura, M. Hattori, and S. Papadimitriou, "Berth allocation at indented berths for mega-containerships", *European Journal of Operational Research*, vol. 179, no. 2, pp. 579 – 593, 2007.

H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling", *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 204 – 223, 2003.

M. Jahangirian and G. V. Conroy, "Intelligent dynamic scheduling system: The application of genetic algorithm", *Integrated Manufacturing Systems*, vol. 11, no. 4, pp. 247 – 257, 2000.

P. Jaillet, "Stochastic routing problems", *Stochastics in Combinatorial Optimization*, G. Andreatta, F. Mason, and P. Serafini (Eds.), World Scientific, New Jersey, 1987.

P. Jaillet and A. R. Odoni, "The probabilistic vehicle routing problem", *Vehicle Routing: Methods and Studies*, B. L. Golden and A. A. Assad (Eds.), North-Holland, Amsterdam, 1988.

I. Kacem, S. Hammadi, and P. Borne, "Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems", *IEEE Transactions on Systems, Man and Cybernetics, Part C*, vol. 32, no. 1, pp. 1 – 13, 2002.

R. M. Karp, "Reducibility among combinatorial problems", *Complexity of Computer Computations*, Plenum press, New York, pp. 85 – 104, 1972.

S. A. Kazarlis, S. E. Papadakis, J. B. Theocharis, and V. Petridis, "Microgenetic algorithms as generalized hill-climbing operators for GA optimization", *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 3, pp. 204 – 217, 2001.

T. A. Keller and G. B. Lamont, "Optimization of a quantum cascade laser operating in the terahertz frequency range using a multiobjective evolutionary algorithm", in *Proceedings of the 17th International Conference on Multiple Criteria Decision Making*, *MCDM 2004*, vol. 1, 2004.

G. Kendall and N. M. Hussin, "An Investigation of a tabu search based hyperheuristic for examination timetabling", in *Proceedings of Multidisciplinary International Scheduling: Theory and Applications*, *MISTA 2003*, Nottingham, UK, pp. 309 – 328, 2003.

G. Kendall and N. M. Hussin, "A tabu search hyper-heuristic approach to the examination timetabling problem at the MARA university of technology", in *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2004*, Pittsburg, PA, USA, E. K. Burke and M. Trick (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 3616, pp. 199 – 218, 2005.

K. H. Kim and K. C. Moon, "Berth scheduling by simulated annealing", *Transportation Research Part B: Methodological*, vol. 37, no. 6, pp. 541 – 560, 2003.

M. R. Knarr, M. N. Goltz, G. B. Lamont, and J. Huang, "In situ bioremediation of perchlorate-contaminated groundwater using a multi-objective parallel evolutionary algorithm", in *Proceedings of the 2003 Congress on Evolutionary Computation*, *CEC 2003*, Canberra, Australia, vol. 3, pp. 1604 – 1611, 2003.

J. D. Knowles and D. W. Corne, "The Pareto archived evolution strategy: A new baseline algorithm for multiobjective optimization", in *Proceedings of the 1999 Congress on Evolutionary Computation*, *CEC 1999*, Washington, DC, USA, pp. 98 – 105, 1999.

J. D. Knowles and D. W. Corne, "Approximating the nondominated front using Pareto archived evolutionary strategy", *Evolutionary Computation Journal*, vol. 8, no. 2, pp. 149 – 172, 2000.

R. Kumar and P. I. Rockett, "Improved sampling of the Pareto-front in multiobjective genetic optimizations by steady-state evolution: A Pareto converging genetic algorithm", *Evolutionary Computation Journal*, vol. 10, no. 3, pp. 283 – 314, 2002.

K. K. Lai and K. Shih, "A study of container berth allocation", *Journal of Advanced Transportation*, vol. 26, pp. 45 – 60, 1992.

V. Lambert, G. Laporte, and F. V. Louveaux, "Designing Collection Routes through Bank Branches", *Computers and Operations Research*, vol. 20, pp. 783 – 791, 1993.

G. Laporte and F. V. Louveaux, "The integer L-shape method for stochastic integer problems with complete recourse", *Operations Research Letters*, vol. 13, pp. 133 – 142, 1993.

G. Laporte and F. V. Louveaux, "Solving stochastic routing problems with the integer L-shaped method", *Fleet Management and Logistics*, T. G. Crainic and G. Laporte (Eds.), Kluwer Academic Publishers, Boston, pp. 159 – 167, 1998.

R. C. Larson, "Transportation of sludge to the 106-mile site: An inventory routing algorithm for fleet sizing and logistic system design", *Transportation Science*, vol. 22, pp. 186 – 198, 1988.

L. H. Lee and E. P. Chew, "A simulation study on sampling and selecting under fixed computing budget", in *Proceedings of the 2003 Winter Simulation Conference*, New Orleans, LA, USA, 2003.

L. H. Lee, K. C. Tan, K. Ou, and Y. H. Chew, "Vehicle capacity planning system: a case study on vehicle routing problem with time windows", *IEEE Transactions on Systems, Man and Cybernetics, Part A*, vol. 33, no. 2, pp. 169 – 178, 2003.

C.-L. Li, X. Cai, and C.-Y. Lee, "Scheduling with multiple-job-on-one-processor pattern", *IIE Transactions*, vol. 30, pp. 433 – 445, 1998.

A. Lim, "The berth planning problem", *Operations Research Letters*, vol. 22, no. 2 – 3, pp. 105 – 110, 1998.

V. Lotfi, and R. Cerveny, “A final exam-scheduling package”, *The Journal of the Operational Research Society*, vol. 42, no. 3, pp. 205 – 216, 1991.

L. T. G. Merlot. N. Boland, B. D. Hughes, and P. J. Stuckey, “A hybrid algorithm for the examination timetabling problem”, in *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2002*, Gent, Belgium, E. K. Burke and P. De Causmaecker (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2740, pp. 207 – 231, 2003.

Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd Ed., Springer, 1999.

M. Middendorf, B. Scheuermann, H. Schmeck, and H. ElGindy, “An evolutionary approach to dynamic task scheduling on FPGAs with restricted buffer”, *Journal of Parallel and Distributed Computing*, vol. 62, no. 9, pp. 1407 – 1420, 2002.

M. Miwa, T. Inoue, M. Matsuzaki, T. Furuhasi, and S. Okuwa, “Nurse scheduling system using bacterial evolutionary algorithm hardware”, in *Proceedings of the 28th Annual Conference of the Industrial Electronics Society 2002*, *IECON 02*, vol. 3, pp. 1801 – 1805, 2002.

N. Mladenovic and P. Hansen, “Variable neighbourhood search”, *Computers and Operations Research*, vol. 24, no. 11, pp. 1097 – 1100, 1997.

A. Moraglio, Y.-H. Kim, Y. Yoon, B.-R. Moon, and R. Poli, “Generalized cycle crossover for graph partitioning”, in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, *GECCO 2006*, Seattle, Washington, USA, Keijzer *et al*. (Eds.), ACM Press, pp. 1421 – 1422, 2006.

P. Moscato and M. G. Norman, “A `memetic' approach for the travelling salesman problem - implementation of computational ecology for combinatorial optimisation on message-passing systems”, in *Proceedings of the International Conference on Parallel computing and Transputer Applications*, IOS Press (Amsterdam), 1991.

Z. Naji Azimi, “Comparison of metaheuristic algorithms for examination timetabling problem”, *Applied Mathematics and Computation*, vol. 16, pp. 337 – 354, 2004.

Z. Naji Azimi, “Hybrid heuristics for examination timetabling problem”, *Applied Mathematics and Computation*, vol. 163, pp. 705 – 733, 2005.

E. Nishimura, A. Imai, and S. Papadimitriou, “Berth allocation planning in the public berth system by genetic algorithms”, *European Journal of Operational Research*, vol. 131, no. 2, pp. 282 – 292, 2001.

C. Oğuz and M. F. Ercan, "A genetic algorithm for hybrid flow-shop scheduling with multiprocessor tasks", *Journal of Scheduling*, vol. 8, no. 4, pp. 323 – 351, 2005.

L. F. Paquete and C. M. Fonseca, "A study of examination timetabling with multiobjective evolutionary algorithms", in *Proceedings of the 4th Metaheuristics International Conference*, *MIC 2001*, Porto, Portugal, pp. 149 – 153, 2001.

L. F. Paquete and T. Stützle, "Empirical analysis of tabu search for the lexicographic optimization of the examination timetabling problem", in *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2002*, Gent, Belgium, E. K. Burke and P. De Causmaecker (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2740, pp. 413 – 420, 2003.

K. T. Park and K. H. Kim, "Berth scheduling for container terminals by using a sub-gradient optimization technique", *Journal of the Operational Research Society*, vol. 53, pp. 1054 – 1062, 2002.

Y.-M. Park and K. H. Kim, "A scheduling method for berth and quay cranes", *OR Spectrum*, vol. 25, pp. 1 – 23, 2003.

M. Pelikan, K. Sastry, and D. E. Goldberg, "Multiobjective hBOA, clustering, and scalability", in *Proceedings of the 2005 Genetic and Evolutionary Computation Conference, GECCO 2005*, Washington, DC, USA, Beyer *et al*. (Eds.), ACM Press, pp. 663 – 670, 2005.

S. Petrovic and Y. Bykov, "A multiobjective optimisation technique for exam timetabling based on trajectories", in *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2002*, Gent, Belgium, E. K. Burke and P. De Causmaecker (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2740, pp. 179 – 192, 2003.

S. Petrovic and E. K. Burke, "University timetabling", in *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*, J. Leung (Ed.), Chapman Hall/CRC Press, Ch. 45, 2004.

S. G. Ponnambalam, N. Jawahar, and B. S. Kumar, "Estimation of optimum control parameters for job shop scheduling", *The International Journal of Advanced Manufacturing Technology*, vol. 19, pp. 224 – 234, 2002.

J. Y. Potvin and S. Bengio, "The vehicle routing problem with time windows - part II: genetic search", *INFORMS Journal on Computing*, vol. 8, no. 2, pp. 165 – 172, 1996.

C. Prins, "Competitive genetic algorithms for the open shop scheduling problem", *Mathematical Methods of Operations Research*, vol. 52, no. 3, pp. 389 – 411, 2000.

C. Prins, "A simple and effective evolutionary algorithm for the vehicle routing problem", *Computers and Operations Research*, vol. 31, no. 12, pp. 1985 – 2002, 2004.

G. T. Pulido and C. A. Coello Coello, "The micro genetic algorithm 2: Towards online adaptation in evolutionary multiobjective optimization", in *2nd International Conference on Evolutionary Multi-Criterion Optimization, EMO 2003*, Faro, Portugal, C. M. Fonseca, P. J. Fleming, E. Zitzler, K. Deb, and L. Thiele (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2632, pp. 252 – 266, 2003.

K. Puljic and R. Manger, "An evolutionary algorithm with repeated mutations for solving the vehicle routing problem", in *Proceedings of the 27th International Conference on Information Technology Interfaces*, pp. 478 – 483, 2005.

R. Qu and E. K. Burke, "Analysing high level heuristics within a graph-based hyper heuristic for exam timetabling problems", School of CSiT, University of Nottingham, Technical Report NOTTCS-TR-2005-3, 2005.

R. Qu and E. K. Burke, "Hybridisations within a graph based hyper-heuristic framework for university timetabling problems", *Journal of the Operational Research Society*, in press.

R. Qu, E. K. Burke, B. McCollum, L. T. G. Merlot, and S. Y. Lee, "A survey of search methodologies and automated system development for examination timetabling", *Journal of Scheduling*, vol. 12, no. 1, pp. 55 – 89, 2009.

N. J. Radcliffe and P. D. Surry, "Formal memetic algorithms", in *Evolutionary Computing: AISB Workshop*, Leeds, UK, T. Fogarty (Ed.), Springer-Verlag Lecture Notes in Computer Science, vol. 865, pp. 1 – 16, 1994.

I. Rechenberg, *Evolutionsstrategie*, Fromman-Holzboog, 1973.

P. Ross and D. Corne, "Applications of genetic algorithms", *AISB Quarterly*, Ed. T. C. Forgarty, vol. 89, pp. 23 – 30, 1994.

P. Ross, D. Corne, and H.-L. Fang, "Improving evolutionary timetabling with delta evaluation and directed mutation", in *Parallel Problem Solving in Nature*, Y. Davidor, H-P. Schwefel, and R. Manner (Eds.), Springer-Verlag, vol. III, 1994.

P. Ross, D. Corne, and H. Terashima-Marin, "The phase transition niche for evolutionary algorithms in timetabling", in *Proceedings of the 1st International*

Conference on the Practice and Theory of Automated Timetabling, *PATAT 1995*, Edinburgh, Scotland, E. K. Burke and P. Ross (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1153, pp. 309 – 324, 1996.

P. Ross, E. Hart, and D. Corne, "Some observations about GA-based exam timetabling", in *Proceedings of the 2nd International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 1997*, Toronto, Canada, E. K. Burke and M. W. Carter (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1408, pp. 115 – 129, 1998.

P. Ross, E. Hart, and D. Corne, "Genetic algorithms and timetabling", *Advances in Evolutionary Computing: Theory and Applications*, A. Ghosh and S. Tsutsui (Eds.), Springer-Verlag, pp. 755 – 771, 2003.

P. Ross, J. G. Marin-Blazquez, and E. Hart, "Hyper-heuristics applied to class and exam timetabling problems", in *Proceedings of the 2004 Congress on Evolutionary Computation*, *CEC 2004*, Portland, OR, USA, vol. 2, pp. 1691 – 1698, 2004.

A. Sahu and R. Tapadar, "Solving the assignment problem using genetic algorithm and simulated annealing", *International Journal of Applied Mathematics*, vol. 36, no. 1, pp. 37 – 40, 2007.

A. Schaerf, "A survey of automated timetabling", *Artificial Intelligence Review*, vol. 13, no. 2, pp. 87 – 127, 1999.

J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms, genetic algorithms and their applications", in *Proceedings of the 1st International Conference on Genetic Algorithms*, pp. 93 – 100, 1985.

N. Secomandi, "Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands", *Computers and Operations Research*, vol. 27, pp. 1201 – 1225, 2000.

N. Secomandi, "A rollout policy for the vehicle routing problem with stochastic demands", *Operations Research*, vol. 49, no. 5, pp. 796 – 802, 2001.

K. J. Shaw and P. J. Fleming, "Genetic algorithms for scheduling: incorporation of user preferences", *Transactions of the Institute of Measurement and Control*, vol. 22, no. 2, pp. 195 – 210, 2000.

K. Sheibani, "An evolutionary approach for the examination timetabling problems", in *Proceedings of the 4th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2002*, Gent, Belgium, E. K. Burke and P. De Causmaecker (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2740, pp. 387 – 396, 2003.

P. Shrivastava and S. L. Dhingra, "Development of coordinated schedules using genetic algorithms", *Journal of Transportation Engineering*, vol. 128, no. 1, pp. 89 – 96, 2002.

A. Sokolov, A. Sanyal, D. Whitley, and Y. Malaiya, "Dynamic power minimization during combinational circuit testing as a traveling salesman problem", in *Proceedings of the 2005 Congress on Evolutionary Computation*, *CEC 2005*, Edinburgh, Scotland, UK, vol. 2, pp. 1088 – 1095, 2005.

M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints", *Operations Research*, vol. 35, no. 2, pp. 254 – 265, 1987.

N. Srinivas and K. Deb, "Multiobjective optimization using non-dominated sorting in genetic algorithms", *Evolutionary Computation Journal*, vol. 2, no. 3, pp. 221 – 248, 1994.

R. E. Steuer, *Multiple Criteria Optimization: Theory, Computation and Application*, John Wiley & Sons, 1986.

W. R. Stewart and B. L. Golden, "Stochastic vehicle routing: A comprehensive approach", *European Journal of Operational Research*, vol. 14, no. 4, pp. 371 – 385, 1983.

C.-T. Su and Y. R. Shiue, "Intelligent scheduling controller for shop floor control systems: A hybrid genetic algorithm, decision tree learning approach," *International Journal of Production Research*, vol. 41, no. 12, pp. 2619 – 2641, 2003.

K. C. Tan, L. H. Lee, Q. L. Zhu, and K. Ou, "Heuristic methods for vehicle routing problem with time windows", *Artificial Intelligence in Engineering*, vol. 15, no. 3, pp. 281 – 295, 2001a.

K. C. Tan, T. H. Lee, K. Ou, and L. H. Lee, "A messy genetic algorithm for the vehicle routing problem with time window constraints", in *Proceedings of the 2001 Congress on Evolutionary Computation*, *CEC 2001*, Seoul, Korea, vol. 1, pp. 679 – 686, 2001b.

K. C. Tan, T. H. Lee, and E. F. Khor, "Evolutionary algorithm with dynamic population size and local exploration for multiobjective optimization", *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 6, pp. 565 – 588, 2001c.

K. C. Tan, T. H. Lee, Y. H. Chew, and L. H. Lee, "A hybrid multiobjective evolutionary algorithm for solving truck and trailer vehicle routing problems", in *Proceedings of the 2003 Congress on Evolutionary Computation*, *CEC 2003*, Canberra, Australia, vol. 3, pp. 2134 – 2141, 2003a.

K. C. Tan, T. H. Lee, Y. H. Chew, and L. H. Lee, "A multiobjective evolutionary algorithm for solving vehicle routing problem with time windows", in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, Washington DC, USA, 2003, vol. 1, pp. 361 – 366, 2003b.

K. C. Tan, E. F. Khor, T. H. Lee, and R. Sathikannan, "An evolutionary algorithm with advanced goal and priority specification for multi-objective optimization", *Journal of Artificial Intelligence Research*, vol. 18, pp. 183 – 215, 2003c.

J. C. Tay and N. B. Ho, "Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems", *Computers & Industrial Engineering*, vol. 54, pp. 453 – 473, 2008.

D. Teodorović and G. Pavković, "The fuzzy set theory approach to the vehicle routing problem when demand at nodes is uncertain", *Fuzzy Sets and Systems*, vol. 82, no. 3, pp. 307 – 317, 1996.

D. Teodorović and P. Lucić, "Intelligent vehicle routing system", in *Proceedings of the IEEE International Conference on Intelligent Transportation Systems*, Dearborn, MI, USA, 2000, pp. 482 – 487, 2000.

H. Terashima-Marin, P. Ross, and M. Valenzuela-Rendon, "Clique-based crossover for solving the timetabling problem with GAs", in *Proceedings of the 1999 Congress on Evolutionary Computation*, *CEC 1999*, Washington, DC, USA, pp. 1200 – 1206, 1999a.

H. Terashima-Marin, P. Ross, and M. Valenzuela-Rendon, "Application of the hardness theory when solving the timetabling problem with GAs", in *Proceedings of the 1999 Congress on Evolutionary Computation*, *CEC 1999*, Washington, DC, USA, pp. 604 – 611, 1999b.

H. Terashima-Marin, P. Ross, and M. Valenzuela-Rendon, "Evolution of constraint satisfaction strategies in examination timetabling", in *Proceedings of the 1999 Genetic and Evolutionary Computation Conference, GECCO 1999*, Orlando, Florida, USA, W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith (Eds.), Morgan Kaufmann Publishers, pp. 635 – 642, 1999c.

S. R. Thangiah, J. Y. Potvin, and T. Sun, "Heuristic approaches to vehicle routing with backhauls and time windows", *Computers and Operations Research*, vol. 23, no. 11, pp. 1043 – 1057, 1996.

J. Thompson and K. Dowsland, "General cooling schedules for a simulated annealing timetabling system", in *Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 1995*, Edinburgh,

Scotland, E. K. Burke and P. Ross (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1153, pp. 345 – 363, 1996a.

J. Thompson and K. Dowsland, "Variants of simulated annealing for the examination timetabling problem", *Annals of Operations Research*, vol. 63, pp. 105 – 128, 1996b

J. Thompson and K. Dowsland, "A robust simulated annealing based examination timetabling system", *Computers and Operations Research*, vol. 25, pp. 637 – 648, 1998.

P. Vajda, A. E. Eiben, and W. Hordijk, "Parameter control methods for selection operators in genetic algorithms", in *Proceedings of the 10th International Conference on Parallel Problem Solving from Nature*, *PPSN X*, Dortmund, Germany, G. Rudolph, T. Jansen, S. M. Lucas, C. Poloni, and N. Beume (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 5199, pp. 620 – 630, 2008.

M. Valenzuela-Rendón and E. Uresti-Charre, "A non-generational genetic algorithm for multiobjective optimization", in *Proceedings of the 7th International Conference on Genetic Algorithms*, pp. 658 – 665, 1997.

D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm research: a history and analysis," Department of Electrical and Computer Engineering, Air Force Institute of Technology, Ohio, Technical Report TR-98-03, 1998.

D. A. Van Veldhuizen and G. B. Lamont. "Multiobjective evolutionary algorithms: Analyzing the state-of-the-art", *Evolutionary Computation Journal*, vol. 8, no. 2, pp. 125 – 147, 2000.

R. Varela, C. R. Vela, J. Puente, and A. Gomez, "A knowledge-based evolutionary strategy for scheduling problems with bottlenecks", *European Journal of Operational Research*, vol. 145, no. 1, pp. 57 – 71, 2003.

P. Vincke, *Multicriteria Decision-Aid*, John Wiley & Sons, 1992.

L. Wang and D.-Z. Zheng, "An effective hybrid heuristic for flow shop scheduling", *The International Journal of Advanced Manufacturing Technology*, vol. 21, no. 1, pp. 38 – 44, 2003.

G. M. White and P. W. Chan, "Towards the construction of optimal examination timetables", *INFOR*, vol. 17, pp. 219 – 229, 1979.

G. M. White and B. S. Xie, "Examination timetables and tabu search with longer-term memory", in *Proceedings of the 3rd International Conference on the*

Practice and Theory of Automated Timetabling, *PATAT 2000*, Konstanz, Germany, E. K. Burke and W. Erben (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 2079, pp. 85 – 103, 2001.

G. M. White, B. S. Xie, and S. Zonjic, "Using tabu search with longer-term memory and relaxation to create examination timetables", *European Journal of Operational Research*, vol. 153, no. 16, pp. 80 – 91, 2004.

D. H. Wolpert and W. G. Macready, "No free lunch theorems for search", Santa Fe Institute, Santa Fe, NM, Technical Report SFI-TR-95-02-010. 1995.

D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization", *IEEE Transactions on Evolutionary Computation,* vol. 1, no. 1, pp. 67 – 82, 1997.

T. Wong, P. Côté, and R. Sabourin, "A hybrid MOEA for the capacitated exam proximity problem", in *Proceedings of the 2004 Congress on Evolutionary Computation*, *CEC 2004*, Portland, OR, USA, vol. 2, pp. 1495 – 1501, 2004.

D. C. Wood, "A system for computing university examination timetables", *The Computer Journal*, vol. 11, no. 1, pp. 41 – 47, 1968.

A. Wren, "Scheduling, timetabling and rostering - a special relationship?", in *Proceedings of the 1st International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 1995*, Edinburgh, Scotland, E. K. Burke and P. Ross (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 1153, pp. 46 – 75, 1996.

W. H. Yang, K. Mathur, and R. H. Ballou, "Stochastic vehicle routing problem with restocking", *Transportation Science*, vol. 34, pp. 99 – 112, 2000.

Y. Yang and S. Petrovic, "A novel similarity measure for heuristic selection in examination timetabling", in *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling*, *PATAT 2004*, Pittsburg, PA, USA, E. K. Burke and M. Trick (Eds.), Springer-Verlag Lecture Notes in Computer Science, vol. 3616, pp. 377 – 396, 2005.

Y. M. Ying, "Berth allocation planning using genetic algorithms", Dept. of Civil Engineering, Faculty of Engineering, National University of Singapore, Singapore, Master's Thesis, 1995.

J. Zhang, Y. Jin, and W. Hu, "A genetic algorithm for smooth scheduling in slotted WDM network", *International Journal of Information Technology*, vol. 12, no. 6, pp. 26 – 34, 2006.

E. Zitzler, “Evolutionary algorithms for multiobjective optimization: Methods and applications”, PhD Thesis, Computer Engineering and Networks Laboratory, Swiss Federal Institute of Technology, Zurich, Switzerland, 1999.

E. Zitzler and L. Thiele, “Multi-objective evolutionary algorithms: a comparative case study and the strength Pareto approach”, *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257 – 271, 1999.

E. Zitzler, K. Deb, and L. Thiele, “Comparison of multi-objective evolutionary algorithms: empirical results”, *Evolutionary Computation Journal*, vol. 8, no. 2, pp. 173 – 195, 2000.

E. Zitzler, M. Laumanns, and L. Thiele, “SPEA2: Improving the strength Pareto evolutionary algorithm”, Computer Engineering and Networks Laboratory, Swiss Federal Institute of Technology, Zurich, Switzerland, Technical Report 103, 2001.

E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Fonseca, “Performance assessment of multi-objective optimizers: an analysis and review”, *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117 – 132, 2003.