EVOLUTIONARY COMPUTING
FOR
ROUTING AND SCHEDULING APPLICATIONS

CHEW YOONG HAN
(B. ENG. (COMPUTER ENGINEERING))

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2006

Acknowledgements

Mostly, I would like to thank my supervisor Dr. Tan Kay Chen for his boundless and instructive help, criticism and patience. Without his supervision and expert knowledge in evolutionary algorithms especially in multiobjective optimization, this thesis would not have been possible. I would like to thank Dr. Lee Loo Hay for his guidance in problem modeling and also for his invaluable help, comments and insights in related scheduling applications. I would also thank the professors in the Department of Electrical and Computer Engineering for their constant academic guidance in many aspects.

My gratitude also extends to the officers in the Control and Simulation labs for providing a good environment for conducting research and development. Thanks also to Mr. Heng Chun Meng and Mr. Khor Eik Fun for numerous discussions along the progress of the research. I would like to thank many colleagues and friends who have provided advice and companionship during my study at the Department of Electrical and Computer Engineering. They have offered assistance in many forms help me to overcome various problems in my studies.

Finally, I would like to thank my family for their encouragement and company. Their strong support has given me confidence and courage to move forward.

Table of Contents

Summary

This thesis investigates the use of evolutionary computing technique for solving a range of multiobjective scheduling and routing problems. The optimization for routing problems can be tricky enough even when only elementary constraints are applied, not to mention if other scheduling and time windows information are included in the problems. The magnitude of difficulty for such problems also grows exponentially when the scales increase. The focus of the proposed evolutionary algorithm in the thesis is to handle concurrently multiobjective optimization for routing and scheduling applications. The outline of the contents is listed in the following paragraphs.

The introduction establishes fundamental ideas for the definition of multiobjective optimization and its key importance in decision making process. The definition of evolutionary algorithm and its comparisons to conventional methods such as integer programming and gradient analysis are included. Definitions and examples of scheduling and routing problems are explained. In-depth elaboration on each concept could be found in other subsequent chapters.

Development of recent techniques applied in evolutionary algorithms and problem solving are presented in the Chapter 2. The discussion starts with the reasons for the popularity of evolutionary algorithms in solving scheduling problems, followed by the challenges that are facing by the practitioners. Many examples of scheduling and routing problems are analyzed and then categorized to

illustrate the current landscape of the research domain. The state-of-art of various facets in evolutionary algorithms such as the representation of problem (encoding), the evolutionary operators and the multiobjective optimization features are presented.

In chapter 3, a transportation model for container movements has been built to solve the outsourcing problem faced by a transportation company. The vehicle routing problem (VRP) models a local logistic company provides transportation service for moving empty and laden containers. A Vehicle Capacity Planning System (VCPS) is implemented by modeling the scenario into a Vehicle Routing Problem with Time Windows constraints (VRPTW). It demonstrates solving real world application by using problem modeling techniques which had then triggered the inspiration for the further research exploration in this thesis.

In chapter 4, the design of an evolutionary algorithm to solve multiobjective vehicle routing problem with time windows (VRPTW) is investigated. The proposed algorithm, Hybrid multiobjective evolutionary algorithm (HMOEA) is elaborated. The results of the benchmark problems are then compared extensively with several others implementations. The focus of solutions is on the importance of providing multiobjective solutions in optimization as compared to single objective approaches. The assessment of results was done by using a set of famous benchmark problems.

Furthermore, the optimization of a real-life vehicle routing system with truck and trailer constraints is analyzed in Chapter 5. A new problem model is proposed and optimized. The results from the optimization provide useful information to logistics management. The HMOEA that caters for this specific problem is presented together with the analysis of the results. The comparisons of the choices of the evolutionary operators are also conducted.

A short conclusion provides the final touch on each topic that has been discussed. It also summarizes and comments on the key points to consider when using evolutionary algorithms in real world applications. Finally, several exciting potential enhancements related to current research topic are briefed in Chapter 7.

List of Tables

List of Figures

List of Abbreviations

ARC	Average routing cost
BB	Block booking
B&B	Branch and bound
COD	Complete of discharge
CR	Cost of routing
DM	Decision maker
DT	Decision tree
EDD	Earliest deadline
ETA	Estimated arrival time
ETD	Estimated departure time
EA	Evolutionary algorithms
FMS	Flexible Manufacturing System
GA	Genetic Algorithms
GPX	Generalized position crossover
HMOEA	Hybrid multiobjective evolutionary algorithm
JSP	Job shop scheduling problem
LP	Linear programming
LTTC	Less trailer test case
MINLP	Mixed integer nonlinear programming problem
MO	Multiobjective
MOGA	Multiobjective genetic algorithm
MOEA	Multiobjective evolutionary algorithm
MOP	Multiobjective problem
MRP	Manufacturing resource planning
NLP	Nonlinear programming
NSGA	Non-dominated sorting genetic algorithm
NPGA	Niched pareto genetic algorithm

NSP	Nurse scheduling problem
NV	Number of vehicles
PE	Elastic rate
PM	Mutation rate
PMX	Partially mapped crossover
PPS	Product planning and scheduling
PS	Squeeze rate
PSA	Pareto simulated annealing
PSGA	Problem space genetic algorithm
RAR	Remove and reinsert
RCPS	Resource constrained problems
RNI	Ratio of non-dominated individuals
SPEA	Strength Pareto evolutionary algorithm
TEPC	Trailer exchange point case
TEPS	Trailer exchange points
TS	Tabu search
TTRP	Truck and trailer routing problem
TTVSP	Truck and trailer vehicle scheduling problem
VCPS	Vehicle capacity planning system
VEGA	Vector evaluated genetic algorithm
VHM	Virtual heterogeneous machine
VRP	Vehicle routing problem
VRPTW	Vehicle routing problem with Time Windows constraints
WIP	Work in progress

Chapter 1 Introduction

The thesis revolves around several keywords which happen to be some lexicon that are common in daily conversations. To ensure the semantics are conveyed precisely in this context, definitions of these words such as optimization, multiobjective, routing and scheduling problems are presented in this chapter. Nonetheless, elaborate discussion of these concepts will be presented in the following chapters respectively.

1.1 Optimization explained

Optimization refers to finding one or more feasible solutions, which correspond to extreme values of one or more objectives. The need for finding such optimal solutions in a problem comes mostly from the extreme purpose of either designing a solution with minimum implementation cost, maximum reliability of system, or any other measurable targets. Optimization methods are of great importance in practice, particularly in engineering problems, scientific experiments and business decision-making. An optimization that involves only one objective function, the task of finding its optimal solution is called single-objective optimization. However, most real world applications involve more than one objective. The presence of multiple conflicting objectives (such as minimizing cost and maximizing reliability) is inevitable in many problems (Deb, 2003). The optimization problems become more interesting when complicated constraints are considered.

1.2 Multiobjective optimization

Multiobjective optimizations tackle more than one objective function at an instant. In most practical decision-making problems, multiple objectives or multiple criteria are evident. Classical approaches solve multiobjective problems by transforming multiple objectives into a single objective and the problems are solved with common single-objective optimization algorithm subsequently. However, there are indeed a number of fundamental differences between the working principles of the single objective optimization versus the multiobjective optimization. In a single objective optimization problem, the task is to find a solution that optimizes the sole objective function. Yet, it is wrong to assume that the purpose of multiobjective optimization is about finding optimal solutions that correspond to each objective function individually.

The principles of multiobjective optimization are closely related to concept of non-dominated solution. A general multiobjective problem (MOP) includes a set of *n* parameters (decision variables), a set of *k* objective functions, and a set of *m* constraints. Objective functions and constraints are functions of the decision variables. The optimization goal is to

Maximize/ Minimize $y = f(\vec{x}) = (f_1(\vec{x}), f_2(\vec{x}), f_3(\vec{x}), ..., f_k(\vec{x}))$

Subject to $\vec{e}(\vec{x}) = (e_1(\vec{x}), e_2(\vec{x}), e_3(\vec{x}), ..., e_m(\vec{x})) \leq 0$

Where $\vec{x} = (x_1, x_2, x_3, ..., x_n) \in X$

$$\vec{y} = (y_1, y_2, y_3, ..., y_k) \in Y$$

Here, $\vec{x}$ is the decision vector, $\vec{y}$ is the objective vector, *X* is denoted as the decision space, and *Y* is called the objective space. The constraints $\vec{e}(\vec{x}) \leq 0$ determine the set of feasible solutions (Deb, 2000).

Without the need of linearly combining multiple attributes into a composite scalar objective function, multiobjective optimization algorithm that incorporates the concept of Pareto's optimality should generate a family of solutions at multiple points along the trade-off surface. The numbers of objectives as well as their interdependence determine the curve shape of trade-off surface.

To illustrate, Fig. 1 shows a general Pareto dominance diagram of a minimization problem with two objectives. Let *A*, *B* and *C* are three feasible solution points while f_1 and f_2 are the objectives in this optimization problem. A feasible solution is Pareto-optimal if, in shifting from point *A* to point *B* in the set, any improvement in one of the objective functions from its current value causes at least one of the other objective functions to deteriorate from its current value (Deb, 1999). Based on this definition, point *C* in Fig. 1 is not Pareto-optimal. Mathematically, an objective vector $\vec{u} = (u_1, u_2, ..., u_k)$ is said to dominate $\vec{v} = (v_1, v_2, ..., v_k)$ (denoted by $\vec{u} \preceq \vec{v}$) if and only if $\vec{u}$ is partially less than $\vec{v}$, i.e., $\forall i \in \{1, ..., k\}$, $\vec{u}_i \leq \vec{v}_i \ \wedge \ \exists i \in \{1, ..., k\} \ : \vec{u}_i < \vec{v}_i$. Let Ω is set of all feasible solutions. A solution $\vec{x} \in \Omega$ is said to be Pareto-optimal if and only if there is no $\vec{x}' \in \Omega$ for which $\vec{v} = F(\vec{x}') = (f_1(x'), ..., f_k(x'))$ dominates $\vec{u} = F(\vec{x}) = (f_1(x), ..., f_k(x))$. The Pareto-optimal set often consists of a family of non-dominated solutions, from which the designer can choose the desired answer depending on his/her preference.

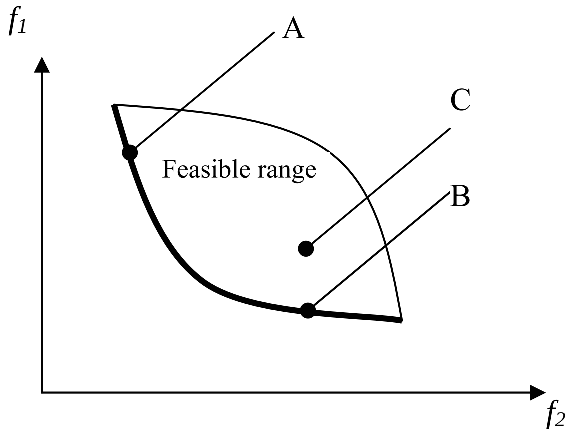


Figure 1 Pareto Dominance Diagram

1.3 Evolutionary algorithms

Evolutionary algorithms (EA) apply the principles of evolution found in nature to the problem of finding an optimal solution. Evolutionary algorithms are global search optimization techniques based upon the mechanics of natural selection and reproduction. They are effective in solving some complex multiobjective optimization problems where conventional optimization tools fail to work well. In "evolutionary algorithms", the decision variables and the evaluation of problem functions are usually direct mappings as contrary to "genetic algorithms" which refer to binary string representation specifically in many literatures. The EA possess an ability to produce robust solutions, because the results of EA are a collection of good traits, which has survived many generations. Evolutionary algorithms for optimization are different from "classical" optimization methods in several ways. A few key concepts that can be found in many variants of evolutionary algorithms are:

- Randomness vs. Deterministic
- Population of candidate solutions vs. Single best solution
- Creating new solution through mutation
- Combining solutions through crossover
- Selecting solutions via “Survival of the fittest”

First, evolutionary algorithms rely in part on random sampling. A nondeterministic method will yield somewhat different solutions on different runs, even if the model has not been changed. In contrast, the linear, nonlinear and integer programming optimization are deterministic methods, they always yield the same solution if simulation start with the same values in the decision variable. This is the characteristic of randomness found in evolutionary algorithm.

Second, where most classical optimization methods maintain a single best solution found so far, an evolutionary algorithms maintain a population of candidate solutions. Only one (or a few, with equivalent objectives) of these are the “best”, but the other individuals of the population are “sample points” in other regions of the search space, where a better solution may later be found. The use of a population of solutions helps the evolutionary algorithms to avoid being "trapped" at a local optimum, when a better optimum may be found outside the vicinity of the current solution.

Third, inspired by the role of mutation of an organism's DNA in natural evolution, evolutionary algorithms periodically make random changes or mutations in one or more members of the current population, yielding a new candidate solution (which may be better or worse than the existing population members). Mutation can happen in many ways. The design of mutation strategy stands an important portion in EA implementation. The result of a mutation operation may be an infeasible solution, and the attempt to repair such a solution to make it feasible is sometime not trivia. Some designers prefer to accept infeasible solutions in the process of simulation and only perform filtering during the final generation.

Another inspiration from the role of sexual reproduction in the evolution of living things, an evolutionary algorithm attempts to combine elements of existing solutions in order to create a new solution, with some of the features of each parent. The elements (e.g. decision variable values) of existing solutions are combined in a crossover operation, as compare to the crossover of DNA strands that occurs in reproduction of biological organisms. There are many possible ways to perform a crossover operation; again this depends on the problem requirement and the representation of problem decision variables in chromosome.

Fifth, inspired by the role of natural selection in evolution, evolutionary algorithms perform a selection process in which the “most fit” members of the population survive, and the "least fit" members are purged. In constrained optimization problems, the notion of "fitness" depends partly on whether a solution is feasible (i.e. whether it can satisfy all of the constraints), and partly on its objective function value. The selection process is the step that guides the evolutionary algorithm towards ever-better solutions.

A drawback of any evolutionary algorithm is that a solution is "better" only in comparison to other, presently known solutions; such an algorithm actually has no concept of an optimal solution**,** or any way to test whether a solution is optimal. (For this reason, evolutionary algorithms are best employed on problems where it is difficult or impossible to test for optimality.) This also means that an evolutionary algorithm never knows for certain when to stop, aside from the length of time, or the number of iterations or candidate solutions, that the user wishes to allow it to explore. Hence, a list of suitable conditions to terminate evolutionary optimizations has also become an exciting research topic itself.

1.4 Scheduling and routing problems

Scheduling aims to determine the sequence of operations. A schedule specifies the operations executing in each step or state. The definition of a schedule is better defined as "A plan of work to be executed in a specified order and by specified times." It can be seen as a plan for performing work and achieving an objective, by specifying the order and allotted time for each part. Baker (1974) defined that a scheduling problem is one which involves "the allocation of resources over time to perform a collection of tasks." The order or the sequence can be the answer to a scheduling problem despite the fact that there are usually related to time unit. To make a schedule is to select jobs or tasks that are to be dispatched.

In forming a complete schedule (such as instructions on a multiprocessor system), two steps occur: sequencing of the jobs and scheduling those prioritized jobs. The distinction between sequencing and scheduling is often not mentioned since the operations are very closely related. They are usually solved concurrently. Hence, general scheduling problems deal with the permutations of a set of jobs, followed by optimizing the placement of these jobs into time slots. Conflicts in resource usage are common observations that prevent a perfect schedule to be arranged. Examples of scheduling problems are evident in all engineering fields, scientific research, and operations research such as: jobs scheduling, resource-constraint project management, nurse scheduling in hospital, crews scheduling for flights, timetable for school and instructions scheduling in parallel computer systems. In summary, all the scheduling problems share a common attribute that deal with time as one of the resources or may be as a variable.

Routing problems are closely related to scheduling and sequencing problem as mentioned above. A the first glance, both the problems belong to combinatorial optimization problems. The solutions with good quality for these problems are usually not easy to obtain. In addition, timing is always an issue in many real world applications for the routing problems. In fact, many routing constraints are imposed due to the time windows constraints. Some of the supplementary scheduling problems such as drivers' scheduling problem and maintenance scheduling problem will also incur additional constraints to the modeling of the routing problems.

1.5 Vehicle routing and applications

In today's business world, transportation cost constitutes a large portion of the total logistics costs. This share has experienced a steady increase, since smaller, faster, more frequent and more reliable transportation are required as a result of trends such as

- Increased variability in consumer's demands
- Quest for quality service management
- Near-zero inventory production and distribution systems
- Sharp global-size competition

The benefit that may be achieved by reducing the transportation costs is of interest to the business at the micro level, and to the country at the macro level. It should come as no surprise that many people in business and researchers in management science and operations research have shown great interest to transportation in the logistics activities. Vehicle routing is the problem of determining the best routes and/or schedules for pickup/delivery of passengers or goods in a distribution system. The objective is to minimize time/monetary/distance measure, given some relevant parameters such as: size of the fleet used by firm, number of drivers, number of routes run daily, inter-city or intra-city operation, total annual cost, crew and vehicle costs. A simple example is to minimize the total distance traveled during delivery of a week’s orders to customers dispersed in a certain geographical region using only one vehicle starts from a central depot. In this example, the distances from the depot to the destination points (customers) and the distances between destination points are the parameters involved.

Some other frequent examples of vehicle routing are:

- Routing of containers among depots, port hubs, warehouses for import and export business activity
- Routing of passenger cars to transport elderly or disabled passengers in a metropolitan.
- Routing of cargo ships to transport loads between seaports
- Routing and dispatching of multi-load vehicles to transport work within processes between workstations in a factory

Routing and scheduling often based on the relative importance of the spatial and temporal aspects of a problem. Classification can be made based on problem models, constraints applied or solution techniques to be used. Typical constraints in vehicle routing might include: vehicle capacity, total time that a vehicle can spend on route and assignment of drivers and other necessary resources such containers and trailers. Several classifications of Vehicle Routing Problems (VRP) are:

- Single Origin-Destination Routing (pure pickup or pure delivery)
- Multiple Origin-Destination Routing (Lim and Fan, W., 2005)
- Single Vehicle Origin Round trip Routing (backhaul)
- Single Vehicle pickup and delivery (Kammarti *et al.*, 2005)
- Other Vehicle Routing and Scheduling

Single origin-destination routing is also known as shortest path problem, and is optimally solvable by Dijkstra's Algorithm (Dijkstra E. W., 1959) if all the transportation costs are nonnegative. Problems up to around 100,000 nodes are solvable in reasonable times using this algorithm. Multiple origin-destination routing is modeled as a network flow problem that can be solved using network simplex algorithm in a reasonable amount of time. Single vehicle-origin round-trip routing is traveling salesman problem, and solved to optimality using specialized branch and bound algorithm. Problems with over 2000 nodes are computationally very time-consuming but are solved reasonably well using heuristic algorithms. The vehicle routing and scheduling category encompasses all other vehicle routing problems that do not belong to the previous four classes. This category constitutes of many practical transportation models that are closer to industry applications. Example of application for vehicle routing can be found in Handa *et al*. (2005), in which a salting route optimization during winter was investigated.

Chapter 2 Recent Developments of Evolutionary Algorithms in Related Problems

In this chapter, a detailed literature review is analyzed and presented. Section 2.1 introduces the application of evolutionary algorithms in scheduling solutions, followed by a very brief history of genetic algorithms since early 70s'. Section 2.2 examines the challenges when finding superior scheduling solutions. In section 2.3, various examples of scheduling problems are categorized based on their applications. The reviews of these scheduling problems are essential due to the fact that the research works that focus solely on multiobjective vehicle routing and scheduling are relatively limited. Naturally, these evolutionary scheduling problems become excellent references to the research topic. In section 2.4, the state-of-art of the real world applications is reviewed. A variety of useful evolutionary operators and the attractive multiobjective feature are presented comprehensively in the section 2.5.

2.1 Evolutionary algorithm in scheduling solutions

Evolutionary algorithms have been reported extensively in many applications. The effort plunged into such research has also increased tremendously in both academic and industry organization. Evolutionary scheduling since then has increased popularity among many other approaches. This observation mainly has to do with the increased difficulty of targeted problems as well as the nature of evolutionary algorithm is suitable to optimize timetabling or scheduling problems.

A Genetic Algorithm (GA) typically embodies a search process that simulates evolutionary process in nature. The technique was first suggested by (Holland, 1973; 1975). The algorithm uses a population of individuals in the evolutionary process. Each solution refers to an individual in the population. The population evolves over generations which are analogous to iteration in program implementation (Glibovets and Medvid, 2003). In each generation, the population will undergo different transformations. The terminologies used for these transformations are mutation and crossover operators. Any individual in the population can be chosen and experiences mutation operation. Alternatively, a new individual can also be created by combining two chromosomes (parents). Such an operator is literally referred as cross over (Chung *et al*., 1997). The least fit individuals of one generation are likely to die off in the next generation. The fittest individuals have the higher chance to be reproduced. The individual is sometime called the chromosome. In the context of scheduling or timetabling optimization the chromosome is usually much complex than a binary string. After a series of improvement in every generation, good solutions can be obtained among the individuals of the final population.

2.2 Scheduling and the challenges

Scheduling is concerned with allocating limited resources to tasks to optimize certain objective functions. On-time delivery of jobs has become one of the crucial factors for customer satisfaction. Scheduling problem is a decision making process. It can have a goal or many objectives (Ponnambalam *et al.*, 2001a). Attempts to optimize scheduling problems have been done using many existing methods. In Fu (2002), an outline of approaches that have been applied to solve several scheduling problems can be seen. They include methods such as: gradient search, random search, simulated annealing, genetic algorithm, Tabu search, neural network and mathematical programming. Many scheduling problems are so complex that they cannot be formulated easily as mathematical programs, (e. g. Integer programming). The fact that they are difficult to formulate makes them tricky to be solved when applying classical techniques such as branch and bound or dynamic programming. (Chung *et al.*, 1997). Scheduling is known to be a hard problem (Wen and Eberhart, 2002) for several reasons as elaborated below.

First, it is a computationally complex problem, which means that search techniques that search the space of solutions deterministically and exhaustively will probably fail to find any solution (if time is limited). In other words, to promise an optimal search using conventional methods can be very expensive. Sometimes, it is like looking a needle in a haystack problem. Second, scheduling problem are often made complicated by the detail of a particular scheduling scenario. Evolutionary algorithms give a considerable flexibility in adapting the techniques to particular application because in most cases, domain knowledge can be managed separately. Third, a solution to a scheduling problem can be deceptively local optima instead of a global best solution. In many cases, exhaustive search is infeasible for NP-complete problems due to the immense search space, it is also difficult to determine whether a solution is local optimal or global optimal. Forth, it is highly constrained in nature; the problem could have no feasible solution not to mention an optimal solution. For example, an examination scheduling problem can be very hard to solve when the examination period is very limited. Numerous modules have to be arranged into different time slots while students usually take more than one modules in one semester. Finally, a scheduling problem becomes huge or can grow to a large problem from a very simple basic model. When this happens, the computation time for solving this problem does not only grow linearly, but exponentially in most case. All the above characteristics explain briefly why scheduling is a difficult problem to solve.

Likewise, sequencing problems are difficult combinatorial problems because of the extremely large search space for possible solutions plus many deceitful local optima can exist. The search space for the sequencing problem can hardly be predictable. Search landscape of a realistic single-machine scheduling task (Darwen, 2002) shows that the near optimal solutions (the best and the second best) have only 56% in common. This indicates that local optimal is very common because when a searching procedure is not able to find any better solution around the neighborhood, it tends to presume that it has made to the global optima.

For instance, creating manufacturing schedules is a critical function in any manufacturing processes nowadays. It is not only about decision making process that deals with resource allocation; it has to ensure the correct timing issues simultaneously (Gürsel *et al*., 2003). The problems are usually highly constrained as resources in real life are always limited. Manufacturers face many challenges when attempting to make decisions faster in large scale scheduling. Besides, the process of scheduling is interweaving with many activities in an organization. In a hierarchical approach, scheduling is usually performed after planning is endorsed at higher level. Manufacturing schedules can then be broken down to the details of every activity; therefore a scheduling horizon is usually shorter than a planning horizon. Such limitation contributes to the difficulty of solving scheduling problems.

Is scheduling a solved problem? Summarizing the current state of the art, many research opportunities are available to improve the scalability and flexibility of scheduling algorithm. Current scheduling techniques are capable of solving large problems (i.e. tens of thousands of activities, hundreds of resources) in reasonable time frames. They are capable of creating schedules under broad and diverse sets of constraints that include time and resource capacity. Research in applying various global, local and meta-heuristic based search frameworks to scheduling problems has produced a number of general approaches to scheduling optimization. Furthermore, increasing integration of AI-based search techniques such as evolutionary algorithm yields more powerful optimization capability.

There have been a number of developments of evolutionary scheduling solutions in literatures. Davis L (1985) is said to be the first to suggest and demonstrate the use of Genetic algorithm (GA) on a simple job shop scheduling problem. Subsequently, many publications investigating on relevant problems are found in journals and conferences. Improvement and successful research reports have been published by researchers all over the world. Some early attempts of solving shop scheduling problem using evolutionary algorithm was mentioned in Varela *et al*. (2003). Dorndorf and Pesch, (1995) studied evolutionary based learning in a job shop scheduling environment. Fang *et al*. (1993) proposed a promising genetic algorithm approach to solve job shop scheduling and open-shop scheduling problems. Syswerda (1991) employed a genetic algorithm to optimize a scheduling problem. Biewirth (1995) proposed a generalized permutation approach to solve scheduling problem and had chosen a job shop scheduling as the example. Yamada *et al*. (1996) published a research that applied a genetic algorithm with hybrid local search and a multi-step crossover. The research presented a job shop scheduling problem as the benchmark for the optimization performance.

Important reviews in the research area are presented in Bruns (1999), Dimopoulos and Zalzala (2000) as well as Burke and Petrovic (2002). Despite the long history of various attempts since 1980-an, most of the job shop scheduling problem reported mainly focused on static scheduling where disturbance does not happen. All operations and machines set were fixed before operation (Chryssolouris and Subramaniam, 2001). A table that summarized several algorithms and their applications on various shop scheduling problem was also presented.

2.3 Scheduling problems in different categories

The machine scheduling can be categorized into single machine problem, parallel machine problem, flow shop scheduling, job shop scheduling, flexible manufacturing system (FMS) scheduling, identical machines scheduling, cellular machines scheduling and so on. The information of the arriving jobs can be deterministic or stochastic. Jobs that only start at time zero are static and jobs that can start anytime are dynamic. Two famous manufacturing shop problems (flow shop and job shop) and floor shop problems specifically FMS problem are reviewed in this section. Research works regarding production planning, and resource constrained planning system are explored. Production scheduling problems together with nurse scheduling problems and other crew scheduling problems are also observed in this section.

In today's complex manufacturing environment, a production site can have several lines running simultaneously, where each requiring different steps and machines for completion. A decision maker for a manufacturing plant needs to find out successful ways to manage various resources so that production can be completed using the most efficient method. The decision maker also needs to create a good production schedule that promotes on-time delivery especially, and minimizes objectives such as the makespan of a product and sometimes the production cost explicitly. Out of these concerns grew an area of studies known as the manufacturing scheduling problems or commonly referred as shop scheduling problems.

Different modes of machine settings are translated into optimization problems. To name a few: single machine model, parallel machine model, flow shop scheduling and also job shop scheduling. Single Machine model is when only one machine is available to process all jobs. Each job has a single task (operation). Every job is performed on the same machine. Parallel machines model consists of multiple machines that are available to process jobs. The machines can be identical, of different speeds, or specialized to only processing specific jobs. Each job has a single operation. The two models are relatively simple compared to those reported in recent literature.

In a flow shop model, there are a series of machines numbered 1, 2, 3…m. Each job has exactly m operations. The first operation of every job is done on machine 1, second operation on machine 2 and so on. Every job goes through all m machines in a unidirectional order. However, the processing time each task spends on a machine varies depending on the job that the operation belongs to. In cases where not every job has m operations, the processing times of the task that do not exist is zero. The precedence constraint in this model requires that for each job, operation (i-1) on machine (i-1) must be completed before the i^{th} operation can begin on machine i.

On the other hand, a job shop model has a set of machines indexed by k. Jobs are indexed by i, and operations are indexed by j. Each operation on a machine is indicated by a set of three indices, i, the job that the operation belongs to; j, the number of the task itself, and k, the machine that this particular operation needs to use. The flow of the operations in a job does not have to be unidirectional. Each job may also use a machine more than once. For example, the following table describes a job shop with two jobs. The entries denote the machine that operation j of job i

needs. For example: Job 1 has only two operations, requiring machine 5 and 6 respectively. Job 2 has three operations, requiring machine 2, 7, and then machine 2 again.

Table 1 Operations in job shop model

JOBS	Op 1	OP 2	OP 3
Job 1	5	6	-
Job 2	2	7	2

* OP stands for operation

2.3.1 Job shop scheduling

Job shop is an NP-hard combinatorial problem (Garey *et al*., 1976; Bruker, 1995). It is therefore unlikely to solve in polynomial time with existing algorithms. Searching the optima answer with branch and bound algorithm approach is possible only for small problems.

Job shop scheduling creates a schedule that defines the time intervals in which the operations are processed, but it is feasible only if it complies with the following constraints: one process at a time for a machine, operation sequence must be respected. No preemption is allowed during the execution. Note that the problem however does not enforce all the jobs to have similar sequence of operations like flow shop problem. Kacem *et al*. (2002a) introduced an evolutionary algorithm hybrid with fuzzy logic that is applied to solve a flexible job shop scheduling problem. In this problem, the schedule needs to organize the execution of jobs on a number of machines. The operations are constrained by precedence and thus non-preemptive. The execution of every job requires a machine.

Carlos A. Brizuela, and Nobuo Sannomiya (2001) investigated a perturbed version of job shop. A framework was incorporated to measure the robustness, diversity of genetic algorithm in solving combinatorial problem. It tried to answer if the tuning of parameter is required if the problem model is slightly changed. Another research by Ponnambalam *et al*. (2002) also contributes to the research about tuning the parameters such as number of generations, probability of crossover and probability of mutation, relating to the problem sizes. Using different control parameters can lead to different optimization results.

Many optimization problems in the industrial engineering world and particularly manufacturing system are difficult to solve by using conventional methods. A modified genetic algorithm for job shop scheduling was developed by Wang and Zheng (2002). The research tried to improve the operators - crossover operator and mutation operator and their research result showed that effectiveness of the algorithm was superior as compared to simple Genetic Algorithm. In addition to that, an effective genetic algorithm for job shop scheduling was developed by Wang and Brunn (2000). Al-Hakim (2001) proposed an analogue genetic algorithm for solving job shop scheduling problem. The algorithm included a new representation and also a way to evaluate the chromosome using the idea from solving analogue circuits.

In Pérez *et al*. (2003), the research focused on finding multiple solutions in job shop scheduling by niching genetic algorithms. Job shop scheduling problem is viewed as a multimodal problem and hence the optimization completes with single solution was not good enough. The research used niching method to in GA to find multiple solutions. Varela *et al*. (2003) used a knowledge-based evolutionary strategy to solve a job shop scheduling problems with bottlenecks scenario. Cheung and Zhou (2001) looked into a unique job shop problem in their research work where setup time before executing the operations is sequence-dependent.

2.3.2 Flow shop scheduling

Flow shop scheduling is one of the best known production shop scheduling problem besides job shop scheduling problem. The problem is a combinatorial optimization problem proven to be NP-complete (Garey and Johnson, 1979). The flow shop problem has n jobs and m machines. As studied by many researchers, it is commonly defined as follows: N jobs is to be processed sequentially on machine 1,…, m. The processing time for every operation of every job on a particular machine is unique and is pre-specified. At any time, each machine can only process at most one job and each job can only be processed on at most one machine. A unique feature in flow shop scheduling is the sequence in which operations are processed is the same for all jobs. The flow pattern (of operations) in every job is fixed. The objective is generally to find out the best permutation so that its makespan (C_{max} = *maximum completion time*) can be minimized. Although all job must have the same operation sequence, some job can just have 0 processing time to indicate that an operation is not required. So, a job can skip particular machine/operation, but the operation sequence must not be violated.

A flow shop problem can have more than one machine. If all the machines have the same job order then it is a "permutation flow shop problem". The problem hence deals with the sequence of processing for a number of jobs order. The operations in a job are going to be processed in the same order using machines or stages, which means precedence is a constraint. In other words, one can observe that the job sequence is similar on every machine. That is, every job has exactly the same operations only then the processing times are different. In summary, the flow shop problem can be defined precisely with 6 criteria: (Ponnambalam *et al*., 2001a)

- Each job has to be processed on all the machines in the order of 1,2,3.. *M* machine. (means the operations must be done sequentially)
- A job consists of multiple operations. There are *J* number of jobs.
- Every machine processes only one job one time
- One job can be processed at one machine one time
- *M* different machines are available continuously starting from time=0
- Every operation must be finished and can not be preempted.

Cavalieri and Gaiardelli (1997) employed two hybrid genetic algorithms for a multiple-objective flow shop scheduling problem where the hybrid genetic algorithms were compared. The first hybrid GA solved an allocation problem followed by sequencing problem of the production lots in a flow shop environment.

In another proposal, GA was hybrid with a dispatching rule. The assignment was done by GA followed by the job sequencing carried out by traditional dispatching rule EDD (earliest deadline). It was a non-linear model and was treated as a multiobjective problem.

An effective hybrid heuristic for flow shop scheduling was also proposed by Wang and Zheng (2003). This publication proposed a hybrid heuristics genetic algorithm to solve a flow shop scheduling problem. The design of the algorithm was the results from a careful investigation on separate components such as the initialization, crossover and mutation operator. Ponnambalam *et al*. (2001a) incorporated a hybrid evolutionary algorithm and conducted a research that was intended to compare existing constructive heuristics and tried to seek improvement from that.

Ishibuchi *et al*. (2003) practiced a much prudent approach when using hybrid algorithm in optimizing flow shop scheduling problem. They investigated the balance between genetic search and local search in memetic algorithms for a permutation flow shop scheduling. A lot-streaming flow shop scheduling was investigated by Yoon and Ventura (2002). In this flow shop problem, a job (lot) was split into a number of smaller sublots such that the job has smaller granularity when it would be processed by machines. Tang and Liu (2002) proposed a modified genetic algorithm for the flow shop sequencing problem to minimize mean flow time, instead of using the popular maximum completion time as an objective function. The flow shop scheduling problem was also attempted and reported by many other researchers. (Burdett and Kozan, 2000; Basseur *et al*., 2002; Zhang *et al*., 2002; Chan and Hu, 2000)

2.3.3 FMS and other shop floor scheduling problems

An FMS (flexible manufacturing system) refers to advanced manufacturing cells that work in group and interconnected to storage system. The system may be controlled by an automated distributed system. The cells are able to identify distinguish different parts processed by the system. They are suitable for quick change to operation instruction and quick change of physical setup.

Hsu *et al*. (2002) applied genetic algorithm to an FMS cyclic scheduling. The research solved a cyclic scheduling problem with respect to many hard constraints, and trying to minimize the Work in Process (WIP). The process flow is similar to flow shop model, but it started and ended at the same operation and hence a cycle was created. Zhao and Wu (2001) made another attempt with FMS problem with multi-route options. This means all the parts types can be processed through alternative routes. There can be several machines for each machine type. The compute time required in finding a solution of a medium size scheduling problem was acceptable.

Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems proposed by Kacem *et al*. (2002b) was different from conventional problem. In which the assignment and scheduling would be combined as a new problem with greater complexity. In another example, an

Intelligent scheduling controller for shop floor control systems was developed by Su Shiue (2003). The control system could be functionally decomposed into two sections, i.e. the planning scheduling and execution of task. Good dispatching rules are required to achieve efficient task scheduling. A hybrid genetic algorithm and decision tree learning approach were integrated and applied in this application. The system identified a set of relevant attributes so that a knowledgebase could be constructed.

Prins (2000) studied an open shop scheduling problem and attempted to solve it with competitive genetic algorithms. Comparing to those job shop and flow shop problem, open shop problem has free job service sequence which leads to a larger solution space. An open shop problem with two machines (or more) and no preemption become a NP-hard problem if one tries to optimize its makespan. Using the open shop problem as the test case, Goh *et al*. (2003) also reported an interesting comparison of several selection operators that could be possibly employed in evolutionary algorithms.

Turkcan and Akturk (2003) proposed a new problem space genetic algorithm (PSGA) to solve a flexible manufacturing scheduling problem. The approach was utilized to generate efficient solutions approximately by minimizing the cost and the total weighted tardiness in the production problem. PSGA is a genetic algorithm coupled with problem specific fast heuristics. The encoding was done at problem space, rather than solution space, so a decoding function needs to be defined. A problem space refers to attributes found in the problem data. For example, in order to solve a task scheduling problem, instead of a job id, it encodes a job's priority (attribute). A decoding heuristic could be devised to build the solution for the schedule. Apart from the works mentioned above, some recent research that tried to solve manufacturing scheduling problems can be found in Hsu *et al*. (2002), Costa *et al*. (2001), Celano (2000), Zhang and Kwon (2001) as well as Dimopoulos and Zalzala (2001).

2.3.4 Production scheduling problem

Scheduling in production is usually more complicated that shop floor problems discussed in previous sections. The problems come in different forms due to diversified categories of products. In this section, a few examples of production scheduling problem are outlined: production planning problems, resource constrained problem (RCPS) and task partitioning problems. The modeling of such scheduling problems usually takes into account many practical considerations in real world. A model is specific to certain scenario. Various models may be required to handle the production scheduling problems. Evolutionary algorithm provides one good feature in this situation as it can be easily tailored or even applied without much modification to optimize the different models. Contrastingly, conventional enumerative mathematical approaches may require more adaptation when solving problems with unique modeling.

Luo and Guignard-Spielberg (2001) solved a problem known as product planning and scheduling in batches (PPS) using evolutionary algorithm. The problem tried to minimize the sum of production, reservation, setup, inventory and shortage costs. The problem was modeled as an MIP problem and then solved by using a hybrid method combining genetic algorithm, linear programming, and ordinal optimization concept. Another research problem (Li *et al*., 2000) was related to planning and scheduling framework in an industrial manufacturing system. This problem is an Earliness/tardiness production planning and scheduling problem ETPSP which combined "Just-in-time" concept to existing MRP (manufacturing resource planning). The integration had of course increased the complexity of the problem. An algorithm was also proposed to consider lot-size optimization in their research.

Baek and Yoon (2002) conducted research in optimizing a dispatching policy in an interconnected, multi-machine system. The problem is a challenging task since it is combinatorial and the jobs were stochastic. A "fuel-sender manufacturing" system was studied, where the facility produces fuel senders systems for passenger cars and light trucks. The assembly manufactures had a total of 41 different products over its 3 manufacturing lines. The proposed algorithm had to cope with the high variability of various products.

Hindi *et al*. (2002) investigated a resource constrained problem (RCPS) which was the optimization of a single mode, single project, resource constrained project scheduling. A project consists of one set of tasks that requires specific time and resources to complete. The objective of the RCPS problem was associated to cost effectiveness. Common constraints were precedence of tasks and as well as the capacity of resources. No preemption was allowed in such problem because interruption could result in higher cost. Meanwhile, Hartmann (2001) addressed a resource constrained project scheduling problem with multiple execution modes for each of the activity. The objective in this problem was still to reduce the makespan in total. This was quite similar to the flexible manufacturing system as compare to conventional manufacturing job floor problem. In the multi-mode RCPS, the activities in project could be performed in multiple ways, especially with the renewable resource such as man power and machine processing. The comparison with other metaheuristics, such as a local search and a truncated branch and bound (B&B) showed that the proposed GA led to optimality of 98.1%. Besides in some cases, when the B&B was not able to find a good solution within 125 seconds (when it was truncated), GA generally always found good enough feasible solutions. Kohlmorgen *et al*. (1999) gave another example of solving RCPS problem using evolutionary algorithm.

Task partitioning (task matching) and task scheduling are required in many applications such as examples in industrial manufacturing co-design systems, parallel processor systems and programmable systems. Sub-tasks (determined from design specification) should be placed in the right place (which means using the right resource) and starts running at the right time (scheduling). Task matching and scheduling are investigated in the following literatures (Dhodhi *et al*., 2002; Zhong *et al*., 2000; Wiangtong *et al*., 2002). Some other relevant researches in production planning and scheduling application using evolutionary algorithms can be found in Liu and Wu (2003), Feldmann and Biskup (2003), Middendorf, *et al*. (2002), Morad and Zalzala (1999) and Borgulya (2002).

2.3.5 Crew scheduling

The driver scheduling problem solved by conventional greedy heuristics can be slow and accuracy was sacrificed for to reduce the runtime required. Li and Kwan (2001) proposed a simulated evolution algorithm (mimics to GA but with only one solution will go to next generation), to solve the problem. The evaluation was done with the help of fuzzy sets theory. The bus and train driver scheduling problem can be viewed as a set covering problem. A hybrid genetic algorithm for scheduling bus and train drivers was developed in Kwan *et al*. (2000). The research introduced the scheduling of drivers for bus and train by using a system based on evolutionary algorithm named TRACS II. The problem had to be modeled into set covering problem before solving using GA. The solution was assisted with column generation process. In addition, a multiobjective metaheuristic for bus driver scheduling problem was studied in Lourenço *et al*. (2001). The three problems mentioned above related to crew scheduling applications are very distinctive from each others. The variables and the constraints of the problems have very little in common. This is apprehensible as the scenarios in such transportation problems were unique to the local geographical conditions of the cities.

2.3.6 Nurse scheduling

Nurse scheduling problem (NSP) is sometimes known as nurse rostering problem. The nurses are to be assigned to different shifts (day, night, mid-night and others) across the planning horizon which is usually one week. There can be as much as 6 shifts in some instances. The head nurse is the person responsible for the task generally. The solution space is highly constrained with the rigid rules in place by the medication boards. Kawanaka *et al*. (2001) pointed that the schedule which was generated using penalty function could end up with an infeasible schedule and violated labor union act sometimes. The work developed an encoding scheme and proposed the design operators that are suitable for solving NSP problem.

Nurse scheduling is a combinatorial problem that its optimal solution is difficult to locate. In addition to that, this problem itself tends to be large. This problem was famous for its complexity and for the number of constraints it associated with. For example: If there are 3 shifts each day (day time, night time and midnight shift). The nurse who has been working for night shift can not be allocated for midnight shift. Besides, a nurse can not be assigned to midnight shift continuously for more than four days. Miwa *et al*. (2002) agreed that the problem was complicated, and the search space was huge where enormous computations were required. The conventional way of doing it manually was not efficient as users must set all the rules before using the system and the result may not as good when local search algorithm was used. Burke *et al*. (2001a) too showed that construction of nurse scheduling problem can be immensely difficult since rostering in health care was usually highly constrained. The research concentrated on experiments to solve the problems using Tabu search, memetic algorithm with steepest descend and finally memetic algorithm with hybrid Tabu search. Burke *et al* (2001b) explained in great details on nurse scheduling problem and provided many suggestions about the modeling of NSP in order to achieve a faster way to evaluate a roster (schedule). This is important in an evolutionary algorithm, because a large portion of computation time for evolutionary optimizations goes to fitness evaluation. Jan *et al*. (2000) and Inoue *et al*. (2003) are two other research works that saw the growing demand for developing practical nurse scheduling solutions.

2.3.7 Power maintenance problem (hydrothermal scheduling)

A thermal generator maintenance scheduling problem is a complex combinatorial problem. The schedule defined the power output for every generator including the time that it must cease to maintenance. X_i was the current maintenance staring period for unit i. C_i was the maximum power capacity that it can produce. There was a constraint for minimum total power generated by the station. In spite of that, all the units need to be scheduled for maintenance according to their usage pattern. Burke and Smith (2000) showed that the problem could be solved using many optimization methods such as linear programming model, dynamic programming, simulated annealing, and genetic algorithms.

El-Sharkh and El-Keib (2003b) studied the maintenance scheduling of generation and transmission systems by using fuzzy evolutionary programming. The schedule can tell the start time of maintenance for each transmission unit, as well as the generating unit. An evolutionary programming was proposed with a fuzzy model to compare the performance of individuals. A hybrid hill climbing method was incorporated to perform feasibility checking. The problem reported has 33 generating units and 179 transmission lines. The challenge to find an optimum could be complicated since the unknown parameters were too many such as the load of the system, prices of fuels, maintenance cost, resources and staff availability. In (El-

Sharkh and El-Keib, 2003a), another evolutionary programming-based solution methodology was proposed for the similar problem. The problem was a mixed type minimization problem, because the decision variables consists of continuous (generator output) and discrete integer (maintenance scheduling starting date and duration). In El-Sharkh *et al*. (2003c) a security-constrained generation maintenance scheduling was investigated.

Upon examination of the variety approaches to power maintenance scheduling, Basu (2004) introduced an interactive fuzzy satisfying method based on evolutionary programming technique to solve a short term hydrothermal scheduling problem. It was a daily planning problem, in power system operation. Basically, the schedule was required to allocate power generation to multiple units from hydro plants and thermal plants with different fuel cost and emission. The major constraint of the problem was the water flow thorough the hydrothermal station, which must follow a rigid water balance control regulation. And there was a limit for the maximum power capacity in each generation unit. The schedule must specify the power output of each thermal unit at every time slot.

Nidul Sinha *et al* (2003) developed a fast evolutionary programming technique in order to solve a short-term hydrothermal scheduling problem. By using the technique, the solution was nearer to global solution within shorter computational time. Hence, the performance was faster than simple GA, SA and gradient search. The result from the algorithm was also least affected by random initialization according to the experiments that were carried out. In Kim and Ahn

(2001), the research presented a new evolutionary algorithm based on sheep flocks heredity model. The generator maintenance problem could utilize up to 23 generators. Further relevant research works in this field are Lee and Jeong (2001), Manzanedo *et al*. (2001) and Dahal *et al*. (2000).

2.3.8 Other scheduling problems

Deb and Chakroborty (1998) formulated a transit system scheduling problem into a mixed-integer nonlinear programming problem. The problem involves a number of resource and service related constraints such as fleet size, minimum and maximum stopping time and others. Evolutionary algorithm was chosen because it is able to handle complex search space with the nonlinear constraints and also a large number of decision variables. Wen and Eberhart (2002) applied genetic algorithm for a logistics scheduling problem. The research considered a cargo items delivery system which may use helicopters, boats and trucks to move the items from one points (base point) to several different points. (Supply point). Each cargo must be delivered within time windows so that no penalty caused. A simple comparison for different operators was performed for such scheduling problem.

In Roman Nossal (1998), an evolutionary approach to multiprocessor scheduling of dependent tasks was discussed. Multiprocessor systems require efficient scheduling solution to work at an optimized condition so that the expensive price paid for the multiprocessor hardware could be worth it. Fortunately, in this scenario only pre-runtime (not real time) periodic tasks were involved. In Fogel and Fogel (1996), evolutionary programming was incorporated to schedule operations on a suite of heterogeneous computers. The research discussed a process scheduling problem for a Virtual heterogeneous machine (VHM) which can be viewed as a collection of parallel machines. In order to achieve high throughput, load balancing was required. Elrad and Lin (1998) discussed the use of evolution scheduling to solve concurrent scheduling problem where many forms of queues (such as ready queue, blocked queue, normal PC queue and concurrent scheduling queue) existed. In fact, a client-server model technique was developed to tackle the problem.

2.4 Development of real world applications

In Kelly (2002), it was pointed that there was a disconnection between the academic worlds to the practical simulation optimization in commercial. The academic problems tend to model with small number of continuous variables with minimal constraints. It could hardly mean anything to the interest of the practitioner because of relatively small problem size. Most of all, the algorithm in commercial should be flexible and reusable so that it can handle events happened in real life better. Last but not least, it was argued that the commercials perceptive of optimization was more towards improvement for solutions, rather than the optimal answer that was almost nonexistent. A number of attempts to solve real world application explicitly were reported. The application are such as production scheduling at petroleum refinery, bust and train scheduling, table-tennis tournament scheduling, nurse scheduling in hospital and chicken catching scheduling. Generally, such problems portrayed a more vivid picture of the situation in real life.

De Almeida *et al*. (2001) studied a multiobjective fitness evaluation technique and its application to the production scheduling in a existing petroleum refinery. Käschel *et al* (2002) investigated the real time property of an enhanced job shop problem. The optimization delivered promising results such that the algorithm was integrated in to two German business software companies and an engineering company. Another case of real world application with evolutionary scheduling in manufacturing problem can be found in Shackelford and Corne (2001).

Kwan and Mistry (2003) developed a co-evolutionary algorithm for train timetabling. While the train services are operated by a number of franchised independent train operating companies, the tracks and stations are shared and centrally run by a network. The suggested approach had the feature of co-operative co-evolutionary to generate the train timetables automatically. Another similar approach was found in Kwan *et al*. (2000) who developed a hybrid genetic algorithm for scheduling bus and train drivers. The research introduced the scheduling of drivers for bus and train by using a system based on evolutionary algorithm named TRACS II. It had been tested on several train companies and bus operators. One of the UK bus operators which had a size of 10,000 buses adopted the system too. Lourenço *et al*. (2001) also reported a bus drive scheduling problem which was an actual real world model.

Schonberger *et al*. (2000) developed an automated timetable generation for rounds of a table-tennis league. The research studied the scheduling of a non-professional table tennis league. The algorithm had been used for almost 2 years in solving real problem. The flexibility of the algorithm that allows users to input the suspended dates was a very useful feature. Hence the method was generally well accepted by the users. Meanwhile, Burke *et al*. (2001a) took the challenge to attempt a nurse scheduling problem in real hospitals. The construction of nurse scheduling problem can be difficult since the problem in health care rostering was highly constrained. The model was derived from Belgian hospitals. Four real world rostering problems were discussed. The problems had different characteristics and yet the algorithm proposed was able to perform well in all of them. The results hence proved the efficiency of the memetic algorithm empirically.

A very interesting and unusual real world problem was published in Hart *et al*. (1999). It was about a schedule of chicken catching of a chicken processing company. It was another investigation into the success of a genetic algorithm on a real-world scheduling problem. The objective of this problem was to minimize the makespan and the resources required. The daily task including scheduling the squads which did the catching and also need the lorries which delivered birds to the factory. A factory should not be idle at any time and hence the lorries must supply the chicken in a rather constant rate. Interestingly formulated as a constrained scheduling problem and solved using genetic algorithm. This scheduling problem in local chicken factory is a real-life application that produced sensible and dynamic adaptable schedules within a short period of time. The results showed that the GA can successfully produce daily schedules in minutes, and its performance was compatible to those produced manually by expert using a few days time.

Wang *et al*. (2000) developed an online-scheduling of a multiproduct polymer batch plant. The application was a model of MINLP (mixed integer nonlinear programming problem). The quality of solution was comparable to those using mathematical programming. Lee and Jeong (2001) created a daily optimal operational schedule for cogeneration systems in a paper mill. The profitability of the system depended on the efficiency of the schedule, which at the same time need to satisfy thermal and electrical loads. Deb *et al.* (2003) solved a casting sequence scheduling problem in which the orders can have different casting weights. The due dates are an important optimization factor in this problem as it is often one of the issues encountered in foundries. Dahal *et al*. (2001) studied a case study of scheduling storage tanks using a hybrid genetic algorithm. The activities included unloading tanks, filling tanks and emptying tanks in a water treatment facility. Whenever a ship approached the port or jetty must discharge ballast water due to some physical facility constraint. A comparison was made with random search and heuristic method (by current practitioner), proved that GA could find a better schedule. Pendharkar and Rodger (2000) applied genetic search to solve a production and transportation problem at coal mines. It was modeled into NLP nonlinear programming. The system could be used to estimate the operational cost of coal mines in a few states of USA.

2.5 Representation in evolutionary algorithms

The representation of chromosomes plays one of the most important roles in every evolutionary application. The choice of representation fundamentally influences other components in evolutionary operators. A good representation chosen help to ensure the entire search area can be explored as much as possible. Interpretation of a chromosome is a process to generate the actual solution for optimization problems. In this section, we look at three general approaches that had been used in recent research works. Direct representation encodes the solution in a straight forward way (mostly a schedule or a timetable), while indirect representation requires addition steps to generate final solution by interpreting the chromosomes. Learning based genetic search does not store scheduling-related values in chromosomes; instead scheduling problems are solved through evolution of learning process. Fig. 2 shows the categories of techniques for chromosome representations explored in this section.

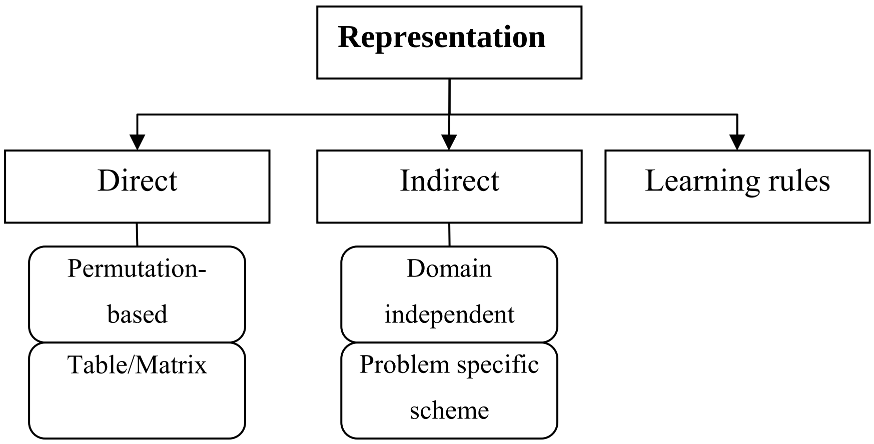


Figure 2 Techniques of chromosome representation

The comparison among different representations was discussed in Xu *et al*., (2005) and Ponnambalam *et al*. (2001b). Ponnambalam *et al*. (2001b) conducted a comparative research using job shop scheduling problem by comparing a number of representation appeared in previous publications. The review showed that finding the best representation were important as the results could be varied significantly even when other factors such as operators, parameter setting and experimental setup were similar. Apart from the quality of solutions, the computational time was another concern. The research repeated several experiments using four different representations, namely the operation based representation, job based representation, preference list representation and priority based representation. The four representations had been widely adopted in many other literatures.

- Operation based representation- encodes a schedule as a sequence of operations. One gene is equal to one operation. In order to preserve feasibility some representations just encode the job numbers and the sequence of operations within the job is arranged according to the precedence constraints.
- Job based representation - allocates resource to first job and then followed by other jobs. Gantt chart can show the schedule clearly. All the operations in the first job are scheduled prior to other operations in other jobs.
- Preference list based - the representation contains sub-chromosome, each for one machine. It does not describe the operation sequence. Instead, it stores the preferred job list of each machine. The decoding procedure is always feasible because it is basically a preference list only.
- Priority rule based - the chromosome encodes dispatching rules for job assignment and the schedule is constructed with the help of heuristics.

GA is used to generate a better sequence of dispatching rule in this representation. The rules can be based on shortest/longest operation time first, shortest/longest processing time, remaining operation, remaining processing times.

2.5.1 Direct representation

Many timetabling systems fall into this category (direct representation) because of assignment to time slots in a timetable can be easily encoded as a matrix with binary values. Permutation of operations with simple repair function is also categorized as direct representation. An issue that always arises in direct representation is the feasibility of chromosomes. The number of rigid constraints is preferably small, since any violation means no feasible solution can be constructed, i.e., when none of possible solutions satisfies all the conditions. If impossibility is referred to hard constraints, the soft constraints would probably be suggesting undesirability. Michalewicz *et al*. (1996) specified that there are seven ways to deal with feasibility of representation: rejection, penalty, repair, replacement with repaired version, use of decoders - such that a chromosome can always be feasible.

2.5.1.1 Permutation based representation

Wang and Zheng (2003) had chosen to use a job permutation to represent a solution when solving flow shop scheduling. An effective genetic algorithm for job shop scheduling was developed by Wang and Brunn (2000). The representation was rather straight forward with a chromosome specified which operation (of which job) was executed at every machine at every moment. Some operations that spanned more than one time slots would be scheduled closely after one another. The validation of chromosome (where it was feasible or not) was examined using a heuristic.

Open shop problems had free job sequences which could lead to a large solution space. In Prins (2000), the representation was nothing new but an ordered permutation of list of operations. However, the chromosome was recovered to active schedules by using three different builders in this case. The research showed that a good builder has significant influence to the optimization results. Middendorf *et al*. (2000) had also chosen to encode a chromosome as a permutation of task number when using evolutionary approach to tackle a dynamic task scheduling problem. While in Tsujimura and Gen (1999), the chromosome was a string of integers (parts' number) that represented a sequence of part loading directly.

2.5.1.2 Table/ Matrix representation

In the attempt to solve a multiobjective evolutionary optimization for flexible job-shop scheduling problems, Kacem *et al*. (2002b) proposed an OMC representation. The problem was different from conventional problem where the assignment and scheduling would be combined as a new problem with greater complexity. The representation of chromosomes was a table with values for each machine (job, operation, start time, end time).

Table 2 The OMC coding

JOB	OPERATION, JOB	M1	M2	M3	M4
J1	O1,1	0	0	0	0, 1
	O2,1	0	0	0	1, 2
	O3,1	3, 6	0	0	0
J2	O1,2	0	0,3	0	0
	O2,2	1,3	0	0	0
	O3,2	0	3, 4	0	0
J3	O1,3	0	0	0, 3	0
	O2,3	0	0	0	3, 4

Hsu *et al*. (2002) focused in solving a cyclic scheduling problem to minimize Work in Process (WIP) with concerns to some hard constraints. The algorithm employed a direct coding using discrete time representation. Each chromosome defined the schedule of operation over a period which was restricted by cyclic time. The chromosome was viewed as a matrix where the elements in the matrix were the operations scheduled to run on each particular machine. The operation was associated with a pair number that tells that which process the operation belonged to, and what time it started on machine. Hence, feasibility was not a problem and no resource conflict could possibly happen. The encoding also required minimum effort in performing interpretation.

A multiobjective metaheuristic for bus driver scheduling problem was presented by Lourenço *et al*. (2001). The chromosome was a binary vector of dimension *n*, indication if the driver duty (column) is assigned in the solution. Sometimes, a greedy heuristic was applied to restore feasibility. Wen and Eberhart

(2002) considered a cargo items delivery system which may use either helicopters, boats and trucks to move the items from one point (base point) to several different points. The cargo item was assigned an ID and the chromosome was an integer string that showed the sequence of the delivery for these cargo items.

While in a nurse scheduling system, Miwa *et al*. (2002) had chosen a chromosome structure that essentially showed the assignment using bit value for each nurse under each possible shift time. For example if there were 20 nurses, 31 days and 4 shifts per day. The chromosome size would be as big as 20 * 31 * 4. (Although the bits can be encoded into fewer bits, this would increase the time required to evaluate the fitness of a chromosome). The representation was as neat as a timetable structure. In a similar nurse scheduling problem, Kawanaka *et al*. (2001) used a representation which was a full schedule for the entire roster, however the schedule had to go through four steps of procedures to ensure the final schedule was free from any violation to hard constraints. There were 6 hard constraints in this case, and basically each of them was taken into consideration sequentially.

In sport events timetabling, different representations were observed. Yang *et al*. (2002) devised a cost effective baseball scheduling by evolutionary algorithms. The base ball sport league schedule arranged the playing team for each time slots (sequentially). The chromosome structure was a sequence of paired number, which one of them was a home team number, another one as a guest team number. Schonberger *et al*. (2000) developed an automated timetable generation for rounds of a table-tennis league. A chromosome was a matrix with row indicated the home team and column was assigned to guest teams. The value insides the matrix were dates (integer) of the games. Repair procedure was outlined to create a complete solution. Despite the repair function, violation of soft constraints would decrease the fitness value via penalty function.

2.5.2 Indirect representation

Indirect representation as mentioned earlier at the beginning of this section does not encode a complete schedule or timetable. Instead, many creative approaches have been adopted in chromosome representation. Sometimes, additional steps are required to interpret a chromosome into a working schedule. Upon examination of the variety of existing approaches, this category is divided into two sub categories based on their representation structures. Domain independent representation has excellent reusability because the scheme was very friendly and easy to adapt. Problem-specific indirect representation does have limited reusability because the chromosomes may not be portable across different applications in scheduling or timetabling.

2.5.2.1 Domain independent (high reusability)

The categories are mainly the priority based representation and some interesting representation that can be applied in other scheduling solutions without much modification. The first two applications in this section have unique representations of chromosomes. The rest are generally using a permutation of priority based representation.

Hindi *et al*. (2002) employed an evolutionary algorithm for resource-constrained project scheduling. Using a permutation number to encode the tasks, a "serial scheduler" was written to schedule each task as early as possible. The serial schedule remained the most important component to achieve high performance in schedule generation. Cowling *et al*. (2002) in the optimization of a trainer scheduling problem had chosen to use a hyperGA approach. By using the hyper GA method, its chromosome had to encode the sequence of performing 12 different low level heuristics to improve the scheduling solution. As a result, the representation became a string of integers stating the sequence for each heuristic method.

Shaw and Fleming (2000), when solving a production scheduling problem, had chosen a chromosome which was a string that encoded the priority of jobs. A schedule builder was required to generate the schedule and evaluation was done based on the schedule produced. The flexibility of such representation was that only minimum alteration on the builder was needed when optimization model had changed. For example, when the production line constraints changed, builder could be adapted according to cater the new requirements. A modified genetic algorithm for job shop scheduling was developed by Wang and Zheng (2002). The representation chosen was a job priority list for every particular time slot, which meant a decoding function would be used to check the feasibility and to build a Gantt chart (schedule). Meanwhile, an analogue genetic algorithm was proposed in Al-Hakim (2001) for solving job shop scheduling problem. Each gene contained the priority for executing the jobs for a particular time segment. Number of genes was equal to number of time segment. Unfinished job operation would have to be extended to next time segment until it was completed. The gene described the priority of job in the time slot. In addition to a specific algorithm that was required to interpret the schedule, a compaction to schedule would be performed before the interpretation.

Rossi and Dini (2000) intended to reduce the production cost by optimizing the performance of a flexible manufacturing system (FMS). A gene encoded the job priority for each machine. The number of genes was equal to the number of machine. In this case, a simple algorithm would interpret the chromosome into a full schedule so that the makespan (objective) can be measured. Esquivel *et al*. (2002) implemented enhanced evolutionary algorithms for single and multiobjective optimization in the job shop scheduling problem. The research had investigated two different indirect representations. The first one was decoder based where the permutation in a chromosome required a builder to generate a schedule. While the priority rule based, a chromosome encoded the dispatching rules so the schedule can be generated using the heuristics. Hartmann (2001) addressed a resource constrained project scheduling problem with multiple execution modes. A chromosome encoded a list of precedence for feasible activities with the mode of operations. (Essentially become paired number- as a gene). Another example was the solution for a pre-cast production scheduling where the chromosome was a string of random values (from 0 to 1) that represented the priority order of building the final schedule (Chan and Hu, 2000).

2.5.2.2 Problem specific scheme

In this section, the chromosomes are usually complicated and do not resemble general data structures. Such encoding schemes are only suitable to use with certain problem domains. When solving a Lecture's scheduling problem, Glibovets and Medvid (2003) had proposed a unique chromosome structure. A gene consisted of 3 pieces of information, (group, room, time) to describe a schedule. The entire chromosome contained static data such as the training course, the lessons plus some dynamic data about the list of students in each group and the teacher of the group. In implementation, a dynamic array was used to encode the whole schedule, which is a set of genes.

Hart *et al*. (1999) also used a special representation when attempted to solve the chicken catching scheduling problem. The chromosome in the application represented the two factories and several strategies for arranging the work orders. In order to decode a chromosome, 4 steps were required: incorporating domain knowledge, sequencing the orders, and splitting and assigning the order. Heuristics were required to help the last two steps. Kacem *et al*. (2002a) introduced an evolutionary algorithm hybrid with fuzzy logic that was applied to solve a flexible job shop scheduling. The representation was a table of assignment to machine plus the starting time and completion time of every job. Another genetic algorithm was applied to solve a production planning and scheduling problem manufacturing system in Li *et al*. (2000). The chromosome was a vector of real values that indicated the planning production quantity for particular products and time slots.

Goh *et al*. (2003) reported a comparison made among several selection methods and the comparison to one proposed method. The comparison was made by using an open shop problem. A permutation of operations was chosen to represent an open job shop solution. Parallel machine tools scheduling problem was studied in Norman and Bean (2000). The problem was solved using a representation known as random keys encoding proposed in Bean (1994). It contained several values from low to high to determine the sequence/ priority when the schedule was being built. A modified genetic algorithm for distributed scheduling problems was reported in Jia *et al*. (2003). The algorithm had to encode the assignment of jobs to different working sites followed by the sequence of operations. As a result, the chromosome grew to a highly complicated structure. Deb *et al.* (2004) proposed a representation using single dimension array to solve a placement problem of electronic components. With this representation formatted, simple cross over and mutation operator can be devised in the evolutionary algorithm.

2.5.3 Learning rules

This section introduces several evolutionary approaches where learning process is the selected tool to solve scheduling problems. Scheduling problems were solved indirectly in such researches by incorporating evolutionary programming or genetics programming. For an instance, dynamic scheduling requires prompt feedback/decision to form schedules for certain jobs/processes. In order to solve a dynamic scheduling problem, a system can learn some reaction rules using evolutionary programming or genetics programming, as these two algorithms are suitable to “generate” programming (knowledge rules) from a pool of literals and operators.

Jahangirian and Conroy (2000) built an Intelligent dynamic scheduling system to generate a robust knowledge about the best dispatching rules for a system where adaptation to dynamic change is a crucial issue. The learning machine was able to generate good solution from initial random one, and the learning could be done incrementally. The solution for a single machine problem with a number of dynamic events source revealed that the system was able to learn and adapt itself in the dynamic environment. Another similar approach was reported in Su and Shiue (2003). An Intelligent scheduling controller was developed for shop floor control systems. The control system can be functionally decomposed into two sections, i.e. the planning scheduling and execution of task. Good dispatching rules were required to achieve efficient task scheduling. GA identified a set of relevant system attributes so that the knowledge base can be constructed. The learning approach combined a decision tree (DT) with genetic algorithm as a hybrid algorithm. Generally a DT algorithm was recruited to build DTs so that evaluation of fitness can be performed.

An intelligent system was created based on an evolutionary knowledge approach (Runarsson and Jonsson, 1999). The ruled based production system was then tested on 10 machines job shops problem. Domain specific knowledge was incorporated into the system without modification to the algorithm itself. Some further research works that employed learning capability of evolutionary algorithm can be found in Aytug *et al*. (1998), Fayech *et al*. (2002), and Podgorelec and Kokol (1997).

A single machine manufacturing problem was presented in Dimopoulos and Zalzala (2001). The research investigated the use of genetic programming for a classic one-machine scheduling problem. Genetic programming was devised to find the set of dispatching rules. The reason of choosing such special representation was to avoid infeasible chromosome which could happen if using conventional permutation encoding. A total of nine dispatching rules would be selected and combined into a set of efficient rules to solve the scheduling problem.

2.6 Crossover operator

Crossover operator is also known as recombination operator. The idea of crossover operation is similar to mating behavior in nature. Generally, two parents are selected from a pool of individuals and new individual can be created by taking information from both of the parents. The interaction can be perceived as an information exchange session among different individuals in a society. Crossover operator has evolved from the traditional one-cut crossover into a variety of interesting procedures today. As evolutionary algorithms are expanding to different areas in engineering optimization, new encoding schemes and crossover operations are being introduced constantly. Choosing a suitable crossover operator is one of the key factors that will determine the quality of the results during optimization (Deb *et al*., 2002; Deb and Beyer, 2001). In this section, five crossover operators are discussed briefly: the order crossover, the cycle crossover, the PMX crossover, the edge crossover and the one-point crossover.

2.6.1 Order crossover

Wang and Zheng (2003) devised a linear order crossover operator. During the operation, two cutting points were chosen, the genes in the cross section were swapped, and the rest of the gene (which did not appear in the cross section) was filled according to the original order of the parent from the beginning. As a result, there would be no redundancy or missing gene. The operation is shown in table 3.

Table 3 Operation of order cross over

STEPS	INDIVIDUAL 1	INDIVIDUAL 2
1. The cut points are shown	26 – 473 - 5891	45 - 218 - 7693
2. Swap the cross section	?? - 218 - ????	?? - 473 - ????
3. Fill the first individual for those does not appear in cross section	64 -218 – 7359 (using the order in parent: 26 - 473- 5891)	?? - 473 - ????
4. Fill the 2nd individual for those does not appear in cross section	64 -218 – 7359	52 - 473 – 1869 (using the order in parent: 45 -218 - 7693)

Another research work that employed linear order crossover was dealing with multiobjective evolutionary optimization for maintenance and production scheduling in job shop problems (Youssef *et al*., 2003). The operation started with exchanging the chunks in between cut points. The outer section was copied from the parent according the method mentioned above. Starting from the beginning of the chromosome, and skipping those which had already appeared in the crossing section. The position of genes mostly would change after the operation, but the orders among the genes were conserved in most occasions. While solving a flow shop scheduling in Ponnambalam *et al* (2001a) used the similar operator with the name as generalized position crossover (GPX) instead. The operator was also found in another scheduling optimization where its representation was a permutation of integers (Shaw and Fleming, 2000). Some other research had selected order based crossover in solving various scheduling and timetabling problems. (Cavalieri and Gaiardelli, 1997; Hussain *et al*., 2002; and Madureira *et al*., 2002)

2.6.2 Cycle crossover

The cycle crossover operator was not as popular as order crossover in scheduling and timetabling optimization although it was widely applied in other applications such as traveling salesman problems by Oliver *et al*. (1987). The cycle crossover is a general crossover operator that preserves the order of sequence in the parent partially. The cycle crossover generates an offspring in which every gene is in the same location as in one parent or the other. This crossover operator tries to avoid cell conflicts by finding non-overlapping sets of genes to pass from the two parents. Its operation is based on the concept of cycle which is a minimal subset of elements such that the set of positions in which they appear are the same in both parents. The details of the operation can be found in Michalewicz (1994). This crossover can be found in Miller *et al*. (1999) where a single machine problem was optimized. The cycle crossover was said to be less positional bias than a normal linear order crossover. A few research works of machine scheduling problem had referred to the cycle crossover. (Hussain *et al*, 2002; Darwen, 2002; Keung *et al*., 2001)

2.6.3 PMX crossover

Partial mapping crossover (PMX), proposed in Goldberg and Lingle (1985), is well accepted among scheduling and timetabling applications. Wang and Zheng (2003) picked this crossover operator in their research of solving flow shop problem. The operators chose two cutting points randomly. This was followed by swapping the chunks between the two parents. The rest of the genes were filled by partial mapping. Repeated genes will be deleted while missing gene was filled as its original order in the parent. In this case, only those in the chunks and repeated genes would have change to new position. The rest just remained as they were:

Table 4 Steps of PMX operator

STEPS	INDIVIDUAL 1	INDIVIDUAL 2
1. The cut points are shown	26 – 473 - 5891	45 - 218 - 7693
2. Swap the cross section	26 - 218 - 5891	45 - 473 - 7693
3. Delete the repeated genes	?6 – 218 – 5?9?	?5 - 473 - ?69?
3. Fill the first individual for missing genes. (according to the sequence in the parent)	46 -218 – 5793 (missing genes are 4, 7 and 3)	?5 - 473 - ?69?
4. Fill the 2nd individual for missing genes. (according to the sequence in the parent)	46 -218 – 5793	25- 473 - 1698 (missing genes are 2, 1 and 8)

Goh *et al*. (2003) had resorted to this operator in their attempt to optimize an open shop scheduling problem. The research made comparison made among several selection methods. To solve an open job shop problem, the chromosome was represented as a permutation of operations. PMX was used to rearrange the permutations as recombination operator. The operator could also be found in a wide range of other applications too. (Hart *et al*., 1999; Hussain *et al*., 2002; Zhang *et al*., 2002)

2.6.4 Edge crossover

Ponnambalam *et al*. (2002) and Whitley *et al.* (1989) were among the researches that used edge crossover. It was simple and the length of chromosome was short enough to be manipulated using edge crossover. This operator split the parent chromosome into two parts with a random cut point from 1 to (m-1). Then, interchange the genes from that crossover position. In Hussain *et al*. (2002), the recombination operator used an edge map to construct the offspring.

2.6.5 One point crossover

An approach to solve the train time table problem as a part of a public transportation problem was presented in Shrivastava and Dhingra (2002). The chromosome was manipulated using one-point crossover which was simple and easy to use since structure of the chromosome was complex enough. In Hsu *et al.* (2002), a chromosome had its structure as a matrix. The one-point crossover actually cut through a horizontal line across the matrix. The operator can also be found in a multiobjective scheduling problem studied by Cavalieri and Gaiardelli (1997). In addition to that, Al-Hakim (2001) used the inspiration from the solutions of analogue circuits to optimize a job shop scheduling problem. The representation could be viewed as a sequence of numbers (job id) structurally. The research also suggested a multi-parent crossover, modified diagonal crossover that would require more than two parents in completing the operation. Despite all the complicated issue in choosing the mating parents, the operation between these pairs of parents was actually a simple one-point cut crossover. Cowling *et al*. (2002) employed a hyperheuristic genetic algorithm to tackle a trainer scheduling problem. A chromosome was a string of integers and one point crossover was perfectly suitable for their application. Another research work that made use of the simplicity of one point crossover was Gürsel *et al*. (2003).

2.7 Mutation operator

Mutation operations take place after crossover is performed in many evolutionary algorithms. The mutation operators permit us to introduce random variations in the solutions and play an important role in the capacity of the GA to diversify the search. The initial aspiration of using mutation is to prevent the falling of all solutions in the population into a local optimum of the solved problem. Yet, mutation rate is usually set to a relatively small number as high mutation probabilities would destroy the convergence behavior of the optimization process. Some popular mutation techniques are introduced here: swap mutation, swift mutation, insertion mutation and ordered based mutation.

2.7.1 Swap mutation

The operator is similar to ordered based mutation. It moves two elements to each other's position. Hence, it is sometime known as an exchange operator. This operator was implemented in Shaw and Fleming (2000) in the attempt of solving a scheduling problem. The representation of chromosome was a permutation of integers. Hence, applying the swap mutation operation only changed the positions of two integers. Sometimes, the operator was known neighbor-swap operator because the small change that it applied was similar to a neighborhood search (Wang and Brunn, 2000). Another instance of such operator was known as exchange operator in a multi-objective evolutionary algorithm that solved a flow shop problem (Basseur *et al*., 2002). Even in a huge scaled problem such as train time table problem (Shrivastava and Dhingra, 2002), swap mutation was chosen as a component in the algorithm. The operation was quite adaptable to different structures of chromosome. The swap mutation was proven a popular choice among the applications of evolutionary scheduling. Mainly because the operation only triggered minimum change to chromosome, which was what the designers are looking for. It also appeared in many literatures such as: Wang and Zheng (2003), Hussain *et al*. (2002), and Zhang *et al*. (2002).

2.7.2 Swift (RAR) mutation

The operation of this mutation is simple. Basically, one element will be removed and reinserted to some random position. In an application of flow shop sequencing, an integer string was chosen to represent the possible solutions. Swift mutation was then applied to mutate the sequence. (Burdett and Kozan, 2000; Puljic and Manger, 2005). The operation was sometimes known as insertion mutation as in Wang and Zheng (2003). The exact procedure of its operation was the same where two elements were chosen randomly. The second element (later as its position is later in the string) was inserted to the position before the first element.

2.7.3 Insertion mutation

Insertion mutation (Basseur *et al*., 2002) can happen in any place. It inserts elements to some random position chosen. This may be similar to RAR mutation only that the insertion mutation may cause shifting position of more than one element in one operation. Hsu *et al*. (2002) incorporated a similar operator when solving F.M.S. cyclic scheduling. Repair or adjustment was required most of the time after the operation. A research about memetic algorithm conducted by Ishibuchi *et al*. (2003) also devised an insertion mutation as one of its evolutionary operators. Youssef *et al*. (2003) solved a production scheduling problem which the lower bound was known. In their implementation, insertion mutation operator was applied and a gene was moved to a new position of other genes, while the remaining genes would be shifted left.

2.7.4 Order based mutation

The operation of order based mutation was similar to swap mutation. The only difference might be the designer perceived that the order of genes in a chromosome was important especially in scheduling problem. The two mutation operations do not demonstrate any significant difference in their implementation. According to Hart *et al*. (1999), the order based mutation swapped two integers in the chromosome.

Ponnambalam *et al*. (2001a) perceived that what the operator did was swapping two gene values, in their application both of the genes were integers. Many other researches had implemented order based mutation in their solutions. (Cavalieri and Gaiardelli, 1997; Ponnambalam *et al*., 2001b; Varela *et al*., 2003)

2.8 Multiobjective research

Examining recent reports, many multiobjective optimization researches are still challenging benchmark problems which optimal solutions are already known. Some problems dealt with simple neighborhood structure at problem space. However, increasing number of research results reporting on multiobjective approach have been observed. (Murata *et al*., 1996) MOEA can deliver good solutions that are as good as single objective optimization when it provides multiple solutions to choose from. Li *et al*. (2000) discussed a planning and scheduling framework in an industrial manufacturing system ETPSP (earliness/tardiness production scheduling and planning). The three objectives are number of unbalancing processes, cost of early production penalties, and the cost of tardy production penalties. The result for the multiobjective genetic algorithm (MOGA) showed that it had better ability to handle multiobjective functions over a simple GA.

Almost all the scheduling and routing problems are multiobjective in nature. The most popular approach in multiobjective solution is using weighted function to aggregate the objective functions because the method was simple and easy to implement. Such algorithm can be adapted from a single objective version too, little change is needed. Apart from that, transformation into single objective optimization problem is easier to develop. The evaluation of performance can be very simple since only one value is observed. All the computational effort can be channeled to the optimization single objective value. However, many other promising methods can also be employed for multiobjective optimization. Pareto ranking method was one of the proven effective alternatives. The section 2.8.1 gives a quick introduction to several research contributions which have significant impact in the multiobjective research area. Section 2.8.2 briefed some multiobjective effort in various problem domains using a vast variety of methods.

2.8.1 Multiobjective evolutionary algorithm

Evolutionary techniques for MO optimization obtain significant attentions from various fields as researchers discover the advantages of their adaptive search capability to optimize for a set of trade-off solutions. As consequences, there have been many survey studies on evolutionary techniques for MO optimization (Fonseca and Fleming, 1993; Coello Coello, 1998; Zitzler, 1999; Van Veldhuizen and Lamont, 2000). Among these, Coello Coello (1998) is one of the most comprehensive surveys that summarized and organized the information on different techniques. The techniques were classified into three main groups based on different implementation strategies in cost assignments and selection methods. These methods include naïve approaches, non-aggregation approaches and Pareto-based approaches. In each group, a fairly detailed implementation of the methods with useful feedback was given.

In the early stage of MO optimisation, multiple objectives are linearly combined into a scalar objective via a predetermined aggregating function to reflect the search for a particular optimum point on the trade-off surface (Jakob *et al*., 1992; Wilson and Macleod, 1993). The trade off curve is only obtained after numerous trials of the weighting components. The drawback of this approach is that the weights are difficult to determine precisely, especially when there is insufficient information or knowledge concerning the optimisation problem. Besides, there are other objective reduction approaches that transform multiobjective problem into simpler problem such as: using penalty functions (Adeli and Cheng, 1994) and constraints method (transform objectives into constraints).

Schaffer (1985) proposed a vector evaluated genetic algorithm (VEGA) that treats the multiple objectives separately in the evolution in order to generate a set of non-dominated solutions in a single run. Although this method is simple to implement, it only manages to find certain extreme solutions along the Pareto trade-offs. Also, the shuffling and merging of all subpopulations in fact attribute to fitness averaging for each of the objective components (Richardson *et al*., 1989). Goldberg (1989) suggested the Pareto-based fitness assignment scheme as a mean of assigning equal probability of reproduction to all non-dominated individuals in a population. The approach has several variants such as the multiobjective genetic algorithm (MOGA) (Fonseca and Fleming, 1993), non-dominated sorting genetic algorithm (NSGA) (Srinivas and Deb, 1994), niched Pareto genetic algorithm (NPGA) (Horn *et al*., 1994), strength Pareto evolutionary algorithm (SPEA) (Zitzler and Thiele, 1999), and others. Murata (1996) applies adapted genetic algorithm (MOGA) to solve flow shop scheduling problem. An enhanced NSGA that incorporates elitism (the preservation of good EA solutions to the next generation) is introduced and been applied to several machine scheduling problems (Tapan P., 1999).

The NSGA-II proposed by Deb *et al.* (2002b) is among the latest successful multiobjective genetic algorithm that has numerous applications ranging from mathematical optimization test problems to real world optimization problems. The algorithm is well known for its efficiency in solving various optimization problems (Deb and Tiwari, 2005) and has also become a benchmark to many researches especially in real-parameters optimization. The algorithm, similar to other Pareto-based evolutionary algorithm, does not require user to decide on the weights for the objectives (Deb, 2001b).

Nevertheless, the application of evolutionary algorithm in routing and scheduling algorithm is not as straightforward as it may seem. Many of the MOEA cannot operate directly on combinatorial problems. Mostly, the research accomplished in many studies concentrates on test problems where the solutions are in the form of real numbers. Moreover, the comparison of efficiency and performance is easier when real value objectives are chosen. These problems may also come with well-structured solution spaces that have friendly neighbourhood relative to combinatorial problems. Various existing evolutionary operators are designed explicitly to perform optimization under specific condition and usually they are not suitable for combinatorial problems such as routing and scheduling optimization. Indeed, careful investigation has to be performed on these researches of MOEA in order to determine useful information that is applicable in routing and scheduling optimization problems.

2.8.2 Multiobjective solution in scheduling

As discussed earlier, many existing methods are available for performing multiobjective optimization. In this section, seven approaches of multiobjective optimization are summarized: weighted function technique, Pareto ranking technique, optimization with alternate generation, fuzzy inference technique, coevolution technique, normalization technique and combination of Pareto and weighted function technique. Each method is also supported with the examples of the relevant applications.

2.8.2.1 Solving using weighted function

Kacem *et al*. (2002a) introduced an evolutionary algorithm hybrid with fuzzy logic to solve a flexible job shop scheduling problem. The algorithm devised a fuzzy multiobjective evaluation stage to evaluate and compare the solutions according to the different objective functions. It computed the weights for each objective and measured the quality of each solution dynamically. The aim was to investigate any possible improvement of the solutions by controlling the direction of searching and hence construct the final solutions closed to the Pareto front. The objective function in the evaluation for selection process was an aggregate of three objectives formulated originally - makespan, workload of the most loaded machine, total workload for machines.

An interesting problem was studied in Hart *et al*. (1999) using weighted function to solve a constraints problem. The fitness of individuals was assigned as 1 / (1+ penalty) in the proposed algorithm. Cavalieri and Gaiardelli (1997) presented a Hybrid genetic algorithm for solving a multiple-objective scheduling problem. The two objectives were the minimum makespan and the minimum tardiness. The weights of the objectives were obtained from domain knowledge. Another similar approach multiobjective research on shop scheduling problem can be found in Ishibuchi *et al* (2003).

In a nurse scheduling problem, nurses are assigned to different work shifts across the planning horizon which was usually one week (Kawanaka *et al*., 2001). The major objectives in NSP were the quality of shifts for each nurse, allocation of holiday to requested day, violation of night shifts assignments and so on. All together 6 objectives function had been identified with assigned weights ranges from 0.1 to 1.0. In this example, the quality of shifts for each nurse was given the highest priority. Consequently, the factor deserved a 1.0 weight so that its impact would be greater than other objective functions. Meanwhile, a process planning and scheduling problem was solved simultaneously in Morad and Zalzala (1999) as both the components (planning and scheduling) were found closely entwined. However, the choices of evaluation became a tricky part as in what criteria should be evaluated for fitness calculation. The objectives had become total number of rejects and total processing cost of the integrated solution. A reject referred to rejection of inferior product. Obviously, additional objective was included to reflect the performance of planning process in the problem. Zhao and Wu (2001) presented another manufacturing related problem, Flexible Manufacturing Systems incorporated simple weighted evaluation function when computing individuals' fitness.

2.8.2.2 Solving using Pareto Multiobjective

A multiobjective optimization with the implementation of user preferences was presented in Shaw and Fleming (2000). The Pareto method gave them more flexibility when solving the problem. Three objectives in the process scheduling problem were the number of job rejected in the schedule, the number of late job, and the variance between finishing times for different production lines. There were a few ways to incorporate user preference in optimization problem.

1) Priori - this method specifies the certainty by fixing the targeted outcome before the optimization starts. It leaves the decision maker with limited choice after the optimization.

2) Interactive - User needs to react to changing situation during the optimization process by constant updating the preference information.

3) Posteriori - The decision maker have the most burden in solving the problem, as the optimization does not incorporate any multiobjective preferences when solving this problem. At the same way, it gives the largest freedom to decision maker.

Comparing to other methods in solving multiobjective optimization problem, multiobjective evolutionary algorithm (MOEA) tends to save computational time as compare to posteriori method, yet allows users the flexibility to determine their preferences than a priori method.

In transportation application, Lourenço *et al*. (2001) studied a multiobjective metaheuristic for bus driver scheduling problem. The bus driver schedule used to be solved using linear programming (LP) with set covering problem model. By using Pareto ranking approach, the algorithm provided multiple good solutions to the decision maker. Apart from these examples, Pareto ranking had been implemented quite often in manufacturing based multiobjective scheduling problem. Some optimization problems were developed to optimize as many as four objectives concurrently. (Basseur *et al*., 2002; Hsu *et al*., 2002; Shaw *et al*., 2000)

2.8.2.3 Solving using alternate generation

A flexible job-shop scheduling problem and an job assignment problem were combined and became a new problem with greater complexity. After the combination, optimization model was introduced to one additional objective. The objectives were then consisted of makespan and total workload of the machines. Since the operations could be assigned to different machines and may take different processing time to finish (Kacem *et al*., 2002b). The two objectives optimization was achieved by using one objective at a time (alternately every generation). Subsequently in practice, the performance measure was the sum of machines' workload when generation number was even and otherwise it was the makespan when generation number was odd. The result of multiobjective optimization in this case, had found two solutions, which one of them was as good as the solution provided by single objective optimization using makespan as the objective. The other solution found in the research had a shorter makespan but higher machine load.

2.8.2.4 Solving using fuzzy inference

An evolutionary programming technique with fixed coding scheme was presented for a multiobjective short-term hydrothermal scheduling problem in Basu (2004). The work introduced an interactive fuzzy satisfying method based on evolutionary programming technique. The schedule must specify the power output for each thermal unit at every time slot. The main objectives were the cost of operation and the emission level from thermal unit. As the weights would not be determined easily and the two objectives were using different scales (units), it was not suitable to evaluate them using Pareto concept. As a result, the research chose to use a fuzzy satisfying method. The decision maker (DM) can choose the membership function to be used during evaluation process. If the DM was not satisfied with the result, the membership functions could be modified interactively. This had made the algorithm more flexible and friendly to DM.

2.8.2.5 Solve it using coevolution

This is one of the very creative ways to solve a multiobjective problem. In a real-time dynamic shop floor scheduling (Käschel *et al*., 2002) had implemented an evolutionary algorithm that cater two objectives concurrently. The objectives in the problem were the mean lateness and the mean flow time. In the solution, two populations were created and each of the population only showed its interest to one objective value. Esquivel *et al*. (2002) proposed a coevolutionary approach to tackle a multiobjective job shop scheduling problem. The objectives were three different functions: makespan as in overall schedule completion, earliness as well as completion time for every job. Hence, three populations were utilized to evolve the solutions based on three different criteria until termination condition was met. By then, the three populations would be merged and good solutions would be selected from the pool. This technique may require additional number of populations if the number of objectives is increasing. This might be a concern if computation time is limited.

2.8.2.6 Normalized fitness

Chryssolouris and Subramaniam (2001) tried to design a fair evolution algorithm for multiobjective dynamic scheduling problem. The objectives in this research were average job tardiness and total job cost. The reproduction should be done in a fair way by using normalized fitness. After normalization, all objectives played an equal factor of importance in problem optimization. Unfair weights scaling problem can be eliminated. Yet, consistent reproductive pressure was applied.

2.8.2.7 Combination of Pareto and weights

Turkcan and Akturk (2003) had created a unique multiobjective evaluation approach. The approach was used to find efficient solutions minimizing the cost and the total weighted tardiness in a production problem. The multiobjective problem combined Pareto ranking and the weighted function to generate final fitness. The Pareto ranking was determined using the two objectives mentioned above, and it was then adjusted again using aggregation. In overall, this can give a better reproduction probability to individual who was non-dominated (Pareto ranking), or near to Pareto optimality (weighted function contributed to this). The sharing also ensured that preference would be given to those with less number of neighbors around them. Comparison to other linear weighted function and normalized method showed that, the proposed evaluation method worked better.

The reviews of the components in MOEA showed that the research works invested in evolutionary scheduling are vast and immense. The comprehensive discussion on various usages has provided more perceptive on the challenges and hurdles confronted in evolutionary scheduling problems. Valuable information can be extracted and incorporated directly or indirectly in solving multiobjective routing and scheduling problems.

Chapter 3 Vehicle Capacity Planning System

In this chapter, a VRP problem models a local logistic company provides transportation service for moving empty and laden containers within Singapore. The chapter provides a very concise example for the type of optimization problems that researchers are interested. It also demonstrates an example for solving real world application by using problem modeling techniques. The objectives of the problems are elaborated in the following sections. A simple remark at the end of the chapter explains the motivation inspired by this problem which had triggered further investigation to the research reported in this thesis.

3.1 Introduction

The VRP problem models a local logistic company that provides transportation service for moving empty and laden containers within Singapore. Due to the limited capacity of its own fleet of vehicles, the company cannot handle all the job orders, and have to outsource some orders to other smaller local transportation companies. The current operation of assigning jobs for outsourcing goes through two steps. In the first step, a certain percentage of jobs will be pre-selected for outsourcing according to some simple rules. Then at the second step, the rest of the jobs will be put into an in-house computer system which assigns jobs to its internal fleet of vehicles according to some greedy rules, and the remaining jobs that cannot be served by the internal fleet of vehicles will be outsourced. A Vehicle Capacity

Planning System (VCPS), which models the problem as Vehicle Routing Problem with Time Windows constraints (VRPTW) and Tabu Search (TS) is applied to find a solution for the problem.

3.2 Problems and objectives

Everyday the company receives job orders of container movement for the next day, ranging from importation, exportation to empty container movement. The internal fleet of vehicles is used for handling these orders. However, due to the large number of job orders, most of the time, some of the job orders have to be outsourced to other companies for reasons such as exceeding fleet capacity, low revenue or urgency. The outsourcing decision is made through the following two steps:

- Step 1. Jobs for outsourcing are selected by engineers according to their experience together with some simple rules.
- Step 2. The remaining jobs are put into an in-house computer-aided scheduling system for capacity planning. A very simple rule is used in the system to assign jobs for vehicles, i.e., Earliest-Deadline First. The system will pick up those jobs with earlier deadlines for their internal vehicles, until the fleet reaches its capacity limit, and then the remaining jobs will be assigned for outsourcing.

Usually the capacity planning for step 2 is performed only at the end of the day when most orders have come in. Since most of the transportation companies have certain working hours, it is often unlikely for the company to hold on the planning until all the orders to arrive. In other words, decision must be made before the companies that handle outsourcing job close down at the end of day. Therefore it is important for the management to have rules which guide them on how many jobs they should outsource and how to select those jobs for outsourcing.

The objectives of this study include building a transportation model for the company and find a good solution for the problem. Based on the solution obtained from the model, extract new rules on how to assign jobs for outsourcing. Finally, performance of the new rules with the current rules is compared. The VRP model studied in this section is further improved in the subsequent chapters.

3.3 Major operations

There are three major types of container movement: Importation, Exportation, and Empty Container Movement.

3.3.1 Importation

For importation of laden containers, vehicles pick up containers at the port, and send them to customer warehouses. After discharging in the warehouses, the empty containers are sent to depots. In this model, the whole importation trip is considered as two job orders, i.e., one loaded trip from the port to a warehouse and one empty trip from the warehouse to a depot. Fig. 3 shows the importation process of laden containers.

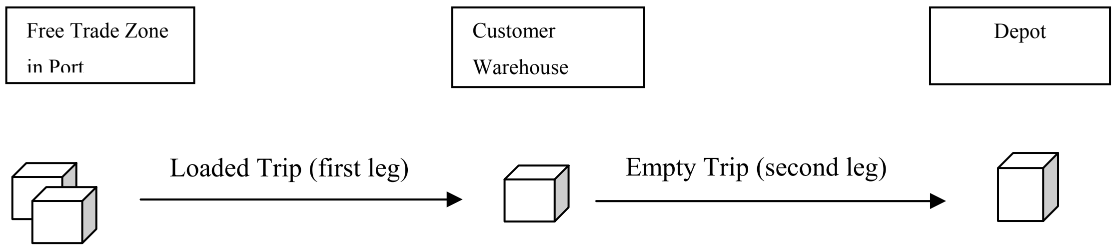


Figure 3 Importation of laden container

Depending on the types of cargoes, each container has different free-storage periods at the port, for example, normal cargo has 72 hours but class 2 cargo (dangerous cargo) only has 24 hours of free-storage time. During this period, vehicles can go into the port at anytime to pick up the loaded containers. Meanwhile, some of the customer warehouses and depots only operate during the usual office hours (i.e., from 8am to 6pm), this time window should also be considered in the model.

3.3.2 Exportation

Similarly, for exportation, the vehicles need to pick up empty containers from depots, and then send them to customer warehouses for loading. After the containers have been loaded, the company needs to book time slots at the port in order to use the crane there to move the containers when they arrive. The time slot given by the port is only 15 minutes and penalty costs are incurred when vehicles do not arrive within the time window. The whole exportation trip will also be considered as two job orders in our model, i.e., one empty trip from a depot to a warehouse and one loaded trip from the warehouse to the port.

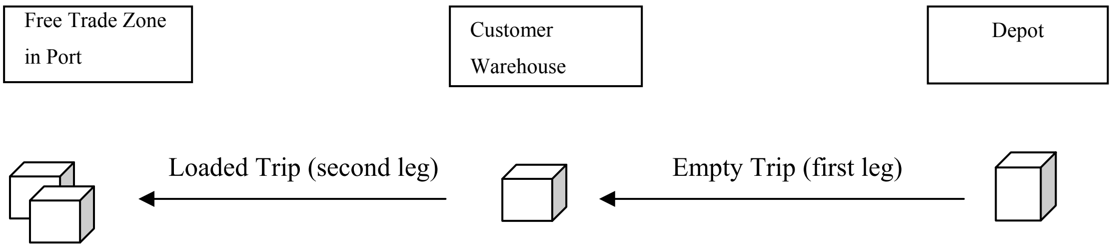


Figure 4 Exportation of laden container

3.3.3 Empty Container Movement

Singapore is the empty container hub for South East Asia, and many shipping liners store their empty containers in the inland container depots in Singapore. Since there is a trade imbalance between different countries, from time to time, the shipping liners need to replenish their containers from one country to another. The empty container movement involves both importation and exportation. For importation activity, empty containers will be picked from the port and sent directly to depots, and for exportation activity, empty containers will be sent directly to the port from depots. Usually, as opposed to other job orders, this type of job orders comes in large quantity. This process is shown in Fig. 5.

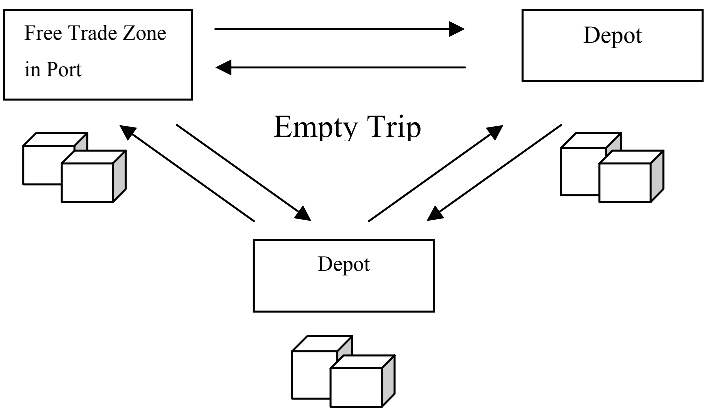


Figure 5 Empty container movement

For importation, the empty containers are taken as normal cargo and enjoy 72 hours free-storage time. For exportation, the port releases a much longer crane-booking time slot to the company, i.e., 4 hours per booking instead of only 15 minutes, and hence the company can move many empty containers into the port at one time. This particular crane booking service is known as Block Booking (BB).

3.4 Problem model

The VCPS problem model is described from the perspective of job details, transportation model and mathematical model.

3.4.1 Job details

In general, when a company receives a job order, it includes the following information:

- Job type (importation, exportation or empty container movement)
- Laden/Empty Trip
- Normal Cargo/Class 2 Cargo
- Trailer type (20 or 40 feet)
- Source/Destination Location
- Handling time in Source/Destination location
- Time windows for Source/Destination location

Time window information of each job is important as it determines the feasibility of the job scheduling. To determine the time windows, we need to know the details of the vessel information, such as Estimated Arrival Time (ETA), Estimated Departure Time (ETD), Complete of Discharge (COD), Ready time, Latest time and Crane booking slot. The time windows vary significantly from type to type, for example, normal cargo importation jobs enjoy time windows of 72 hours at the port, but exportation jobs only have 15 minutes crane booking time slot at the port.

In this study, we divide all the job orders into 7 types, ranging from the importation, exportation to empty container movement

T1) Importation of Normal Cargo from port to warehouse.

T2) Importation of Class 2 Cargo port to warehouse.

T3) Exportation of Normal Cargo from warehouse to port.

T4) Exportation of Class 2 Cargo from warehouse to port.

T5) Importation of Empty Containers from port to depot.

T6) Exportation of Empty Containers from depot to port.

T7) Empty Container Movement from warehouse to depot or depot to warehouse.

3.4.2 Transportation model

The generalized model of a job order can be described in Fig. 6.

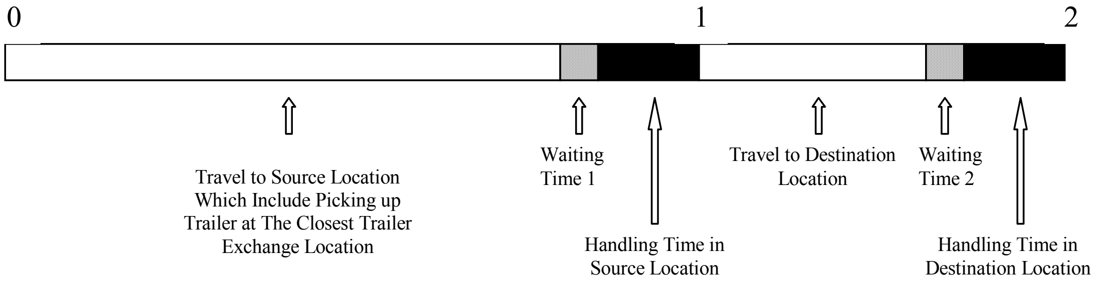


Figure 6 Time sequence of a job model

To process a job order, we first need to travel to the source location of the order with a trailer. Since there is time window constraint in the source location, we might need to wait until the time window is reached, and then the agent at the source location (which can be the port, warehouses or depots) will handle the container and load it to the trailer. Once the container is picked up, it will be sent to the destination location, and the respective agent at the destination location will receive and process the container.

There are two types of containers with two different lengths: 20 feet and 40 feet. Before the trucks go to pick up a container in the source location, it needs to travel to the nearest trailer exchange point to collect the correct type of trailer. Assumption is made that the right type of trailers is always available at every trailer exchange point. In other words, the number of job orders will never exceed the trailer capacity, and hence the trailer type feasibility constraints are not considered in the model. With the knowledge of the location for trailer exchange point, we can always factor in the traveling time to and from the trailer exchange location into the computation of the traveling time from the job starting point to the source location. Although the trailer type does not affect the feasibility of designing a specific route, it contributes to the overall routing performance because the costs of handling different types of containers are different. Under this job model, the vehicle routing and outsourcing assignment problem to be tackled is transformed into a Vehicle Routing Problem with Time Windows (VRPTW) with slight modifications.

As shown in Fig. 7, the VRPTW problem consists of a set of identical vehicles, a set of customer job orders represented by nodes and a network connecting the vehicles and job orders. It is assumed that there are N job orders and K vehicles. Each arc in the network represents a connection between two jobs and indicates the job handling sequence. Each route starts from a truck set-off point, followed by the job orders handled by this truck. The number of routes in the network is equal to the number of vehicles used, and one vehicle is dedicated to one route. Notice that this network does not represent the real geographical connection between job locations. Each job order in the network can be visited only once by one of the vehicles. The time window constraints imposed by each job must be satisfied. Vehicles are also required to complete their individual route within a preset maximal route time, as the drivers have fixed working hours.

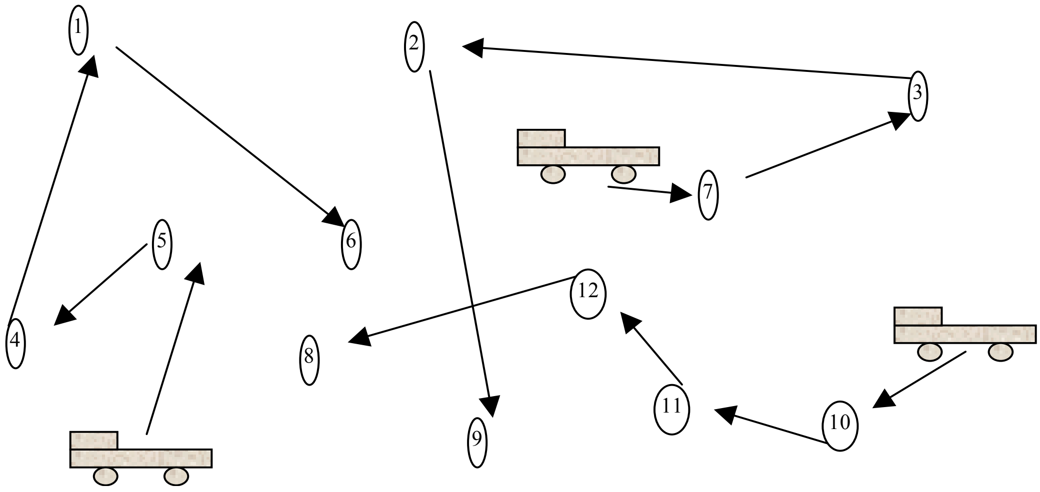


Figure 7 Vehicle routing problem model

3.5 VCPS heuristic

In recent years, a great amount of work has been done on the development of heuristics for the VRPTW problem. Among these methods, Tabu Search (TS) has been shown to achieve significant improvement in optimizing the solutions. TS based on λ-Interchanges is adopted as the method for solving the VRPTW problem. Tabu Search is powerful in searching for solution neighborhood (Chiang and Russel, 1997; Taillard et al., 1997) as compared to other heuristics which may get stuck in local minima.

3.5.1 Initial solution and λ-Interchange Local Search Method

We assume that there are a total of *K* trucks (or *K* routes) and a job pool with all available job orders. To generate the initial solution, we randomly select job orders and insert them sequentially into each route by using the standard Push-Forward insertion method. The Push-Forward insertion method will only allow a job order to be inserted at the place where the feasibility of the route can be maintained. If the job cannot be inserted into the current route, it will be put into a new route. The procedure will continue until no job order can be inserted in any of the K route. All the unassigned job orders are then assigned to truck 0 (or route 0), which represents the subcontractors. There are no time window constraints for this "truck 0". After getting the initial solution, λ-Interchange Local Search Method is used to generate the neighborhood structure. The local search procedure is conducted by interchanging jobs between routes. For a chosen pair of routes, the searching order for the jobs to be interchanged needs to be defined, either systematically or randomly.

3.5.2 Tabu search and heuristic

Tabu Search (TS) uses memory structures to support and encourage a non-monotonic search. Tabu stores the most recent moves or visited solutions in a tabu list. Attempts that reverse the moves or reproduce the solutions in the tabu list will be marked as "Tabu" and be denied. However, an aspiration criterion can release this restriction if a move leads to a new global best solution. The lifetime of a tabu status in the tabu list is controlled by the tabu list size, where First-in-First-out rule is often used for refreshing the tabu list. Structure records the whole route information. For example:

Route 1: $2 \rightarrow 12 \rightarrow 6 \rightarrow 11 \rightarrow 7$

If any of the jobs in this route is removed, the whole route will be recorded as "Tabu". The elements of this structure are strings of job numbers representing recently visited routes. Any future move will be prohibited if it attempts to produce the same route that has been encountered before.

After defining the tabu structure and the local search method, a heuristic is proposed to solve the problem. At the start of the heuristic, an initial solution is generated and then the λ-Interchange Local Search Method is applied to explore the neighborhood of this initial solution. During the search, route 0 will be paired with each route from route 1 to route *K*, and the λ-Interchange operators will examine all the possible moves between each pair of routes that can result in feasible new solutions. The total cost of these newly generated solutions is calculated and put into a candidate list in ascending order.

The move that is ranked first in the candidate list will be checked for validity, i.e., whether it is a "Tabu" or not. If it is not Tabu, this move will be adopted and the solution it produces will be set as the new current solution. After refreshing the tabu list, this iteration is completed. If the first ranked move is Tabu, then the second ranked candidate will be checked until a legal move is found.

3.6 Result and comparison

Altogether 14 test cases have been generated based on statistics provided by logistic company. Seven of the test cases are reserved for rules extraction and another seven for evaluation purpose. After the optimization to selected test cases, the best result obtained at different iteration is shown in Fig. 8. Notice that the costs of the solutions have been normalized with the cost of the final solution obtained. As can be seen, the algorithm is quite effective in improving the solution during the initial phase of the optimization. However, for latter phase of optimization, it has to spend more time to explore the neighborhood in order to escape from local optima.

With the proposed approach, new rules are extracted from the results after the optimization. The new rules are then applied on 7 remaining test cases to evaluate the performance. As a result, the average cost saving (within the capacity limit range of 60-66%) could save up to 8.14%, as compared to the old existing conventional approach used by the logistic company.

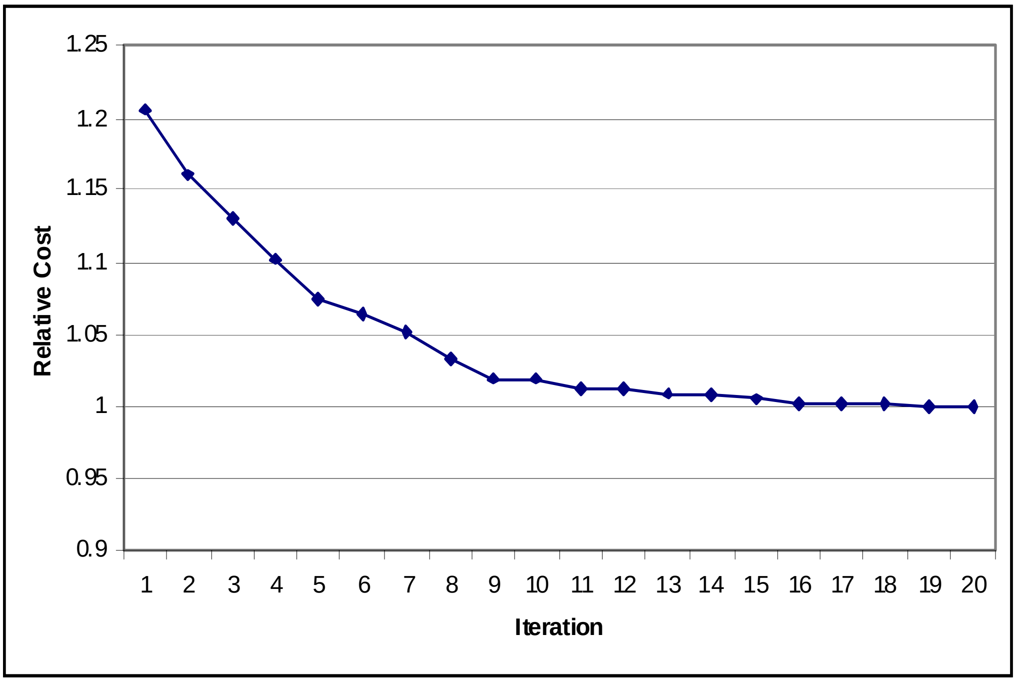


Figure 8 Result of the VCPS

3.7 Remark to research motivation

A transportation model for container movement has been built to solve the outsourcing problem faced by a transportation company. Because of the large amount of job orders, the company must select some jobs to outsource, and the proposed Vehicle Capacity Planning System (VCPS) has helped to select jobs and to minimize the total cost. The transportation model has been built with mathematical definitions, and the advanced artificial intelligence method of Tabu search heuristic has been chosen to solve the problem. This research effort provides strong motivation on further exploring the possibilities of enhancement of the solutions of vehicle routing problems. The optimization in multiobjective perspective for such problem is very useful to logistic operators who strive to reduce their total cost of operations. Consequently, a multiobjective evolutionary algorithm for solving vehicle routing problem is proposed and elaborated. The performance of the proposed algorithm is also examined in the next chapter.

Chapter 4 Hybrid Multiobjective Evolutionary Algorithm for Vehicle Routing Problem

Vehicle routing problem with time windows (VRPTW) involves the routing of a set of vehicles with limited capacity from a central depot to a set of geographically dispersed customers with known demands and predefined time windows. The problem is solved by optimizing routes for the vehicles so as to meet all given constraints as well as to minimize the objectives of traveling distance and vehicles numbers. This section proposes a hybrid multiobjective evolutionary algorithm (HMOEA) that incorporates various heuristics for local exploitation in the evolutionary search and the concept of Pareto's optimality for solving multiobjective optimization in VRPTW. The proposed HMOEA is featured with specialized genetic operators and variable-length chromosome representation to accommodate the sequence-oriented optimization in VRPTW. Unlike existing VRPTW approaches that often aggregate multiple criteria and constraints into a compromise function, the proposed HMOEA optimizes all routing constraints and objectives simultaneously, which improves the routing solutions in many aspects, such as lower routing cost, wider scattering area and better convergence trace. The HMOEA is applied to solve the benchmark Solomon's 56 VRPTW 100-customer instances, which yields 20 routing solutions better than or equivalent to the best solutions published in literature.

4.1 Introduction

In particular, Vehicle routing problem with time window (VRPTW) is an example of the popular extension from VRP. In VRPTW, a set of vehicles with limited capacity is to be routed from a central depot to a set of geographically dispersed customers with known demands and predefined time window. The time window can be specified in terms of single-sided or double-sided window. In single-sided time window, the pickup points usually specify the deadlines by which they must be serviced. In double-sided time window, however, both the earliest and the latest service times are imposed by the nodes. A vehicle arriving earlier than the earliest service time of a node will incur waiting time. This penalizes the transport management either in the direct waiting cost or the increased number of vehicles, since a vehicle can only service fewer nodes when the waiting time is longer. Due to its inherent complexities and usefulness in real life, the VRPTW continues to draw attentions from researchers and has become a well-known problem in network optimization. Surveys about VRPTW can be found in Desrochers *et al*., (1992), Desrosier *et al*., (1995), Golden and Assad (1988), Solomon (1987), Laporte *et al*., (2000), Kilby *et al*., (2000), Toth and Vigo (2002), Bräysy and Gendreau (2001a, 2001b) etc.

A number of heuristic approaches, exact methods, and local searches have been applied to solve the VRPTW which is a NP-hard problem (Beasley and Christofides, 1997; Bräysy, 2003; Breedam, 2001; Chiang and Russel, 1996; 1997; Christofides *et al*., 1981; Desrosier *et al*., 1995; Golden and Assad, 1988; Laporte, 1992; Lee *et* al., 2003; Potvin *et al*., 1993; 1996; Savelsbergh, 1985; Yellow, 1970;

Caseau and Laburthe, 1999; Dullaert *et al*., 2002; Rego, 2001; Bard *et al*., 2002; Gezdur and Türkay, 2002; Ioannou *et al*., 2001; Shaw, 1998; Kilby *et al*., 1999; Li and Lim, 2002; Chavalitwongse *et al*., 2003). While optimal solutions for VRPTW may be obtained using the exact methods, the computation time required to obtain such solutions is often prohibitive and infeasible when the problem size becomes large (Desrochers *et al*., 1992). Conventional local searches and heuristic algorithms are commonly devised to find the optimal or near-optimal solutions for VRPTW within a reasonable computation time (Cordeau *et al*., 2002). However, these methods often produce poor robustness since they could be sensitive to the datasets given. Some heuristic methods even require a set of training data during the learning process, i.e., the accuracy of training data and the coverage of data distribution can significantly affect the performance of the algorithms (Bertsimas and Simchi-Levi, 1993). Such a drawback also becomes apparent when the search space is very large or is unevenly structured for complex VRPTW.

Categorized by Fisher (1995) as the third generation approach for solving vehicle routing problems, evolutionary algorithms (EAs) that emulate the Darwinian-Wallace principle of "survival-of-the-fittest" in natural selection and genetics have been applied to solve the VRPTW with optimal or near-optimal solutions (Gehring and Homberger, 2001; 2002; Grefenstette *et al*., 1985; Homberger and Gehring, 1999; Louis *et al*., 1999; Tan *et al*., 2001a; 2001b; Thangiah *et al*., 1994; Jung and Moon, 2002). Thangiah (1995) proposed a genetic algorithm based approach named GIDEON, which follows the cluster-first route-second method where adaptive clustering and geometric shapes are applied to solve the VRPTW. This approach devised a special genetic representation called genetic sectoring heuristic that keeps the polar angle offset in the genes, and solves the 100-customer Solomon problems to the near-optimal.

Prinetto *et al*., (1993) proposed a hybrid genetic algorithm incorporating 2-opt and Or-opt operations for solving the traveling salesman problem. Blanton and Wainwright (1993) presented two new crossover operators, Merge Cross#1 and Merge Cross#2, and showed that the new operators are superior to traditional crossover operators. Tan *et al*., (2001a) and Thangiah *et al*., (1994) applied hybrid genetic algorithms with Tabu search and simulated annealing for solving the VRPTW and reported some improved routing solutions. Homberger and Gehring (1999) proposed the approach of sub-dividing the optimization problem into phases based on the optimization objectives in VRPTW. In their approach, the optimization was performed in two separate and independent evolution phases, i.e., to minimize the number of vehicles and total traveling distance in the first and second phase, respectively. The parallelization of the metaheuristic was based on the concept of cooperative autonomy, for which several autonomous two-phase metaheuristics cooperate through the exchange of solutions.

The problem of VRPTW involves the optimization of routing multiple vehicles to meet all given constraints. It is required to minimize multiple conflicting cost functions concurrently, such as traveling distance and number of vehicles, which is best solved by means of multiobjective optimization. Many existing VRPTW techniques, however, are single objective-based heuristic methods that incorporate penalty functions or combine the different criteria via a weighting function (Berger *et al*., 2001; Desrosier *et al*., 1995; Golden and Assad, 1988; Toth and Vigo, 2002). Although multiobjective evolutionary algorithms have been applied to solve related combinatorial optimization problems, such as flowshop/jobshop scheduling, nurse scheduling, and timetabling (Ben *et al*., 1998; Burke and Newall, 1999; Chen *et al*., 1996; Murata *et al*., 1996; Jaszkiewicz, 2001), these algorithms are designed with specific representation or genetic operators that could only be used in particular application domains, and cannot be directly applied to solve the VRPTW addressed efficiently.

This research proposes a hybrid multiobjective evolutionary algorithm (HMOEA) that incorporates various heuristics for local exploitation in the evolutionary search and the concept of Pareto's optimality for solving the multiobjective VRPTW optimization. Unlike conventional MOEAs that are designed for parameterized problems (Dias and Vasconcelos, 2002; Cvetkovic and Parmee, 2002; Knowles and Corne, 2000; Tan *et al*., 2001c), the proposed HMOEA is featured with specialized genetic operators and variable-length chromosome representation to accommodate the sequence-oriented optimization in VRPTW. The design of the proposed algorithm is focused on the need of VRPTW by integrating the vehicle routing sequence with the consideration of timings, costs, and vehicle numbers. Without aggregating multiple criteria into a compromise function, the HMOEA optimizes all routing constraints and objectives concurrently, which improves the routing solutions in many aspects, such as lower routing cost, wider scattering area and better convergence trace.

This chapter is organized as follows: Section 4.2 gives the problem formulation of VRPTW, which includes the mathematical modeling and description of Solomon's 56 benchmark problems for VRPTW. Section 4.3 gives a brief description of multiobjectve evolutionary optimization and its applications in a number of domain-specific combinatorial optimization problems. The program flowchart of HMOEA and each of its features including variable-length chromosome representation, specialized genetic operators, Pareto fitness ranking, and local search heuristics are also described in Section 4.3. Section 4.4 presents the extensive simulation and comparison results of the proposed HMOEA based upon the famous Solomon's 56 data sets, which yield 20 routing solutions better than or equivalent to the best-known solutions in VRPTW according to the authors' best knowledge. The advantages of the HMOEA for multiobjective optimization in VRPTW are also discussed in Section 4.4. Conclusions are drawn in Section 4.5.

4.2 The Problem Formulation

This section presents the formulation of the vehicle routing problem with time windows, which involves the routing of a set of vehicles with limited capacity from a central depot to a set of geographically dispersed customers with known demands and predefined time windows. Section 4.2.1 provides the mathematical model of the VRPTW and Section 4.2.2 describes the famous Solomon's 56 benchmark problems for the VRPTW.

4.2.1 Problem Modeling of the VRPTW

This section presents the mathematical model of the VRPTW, including the frequently used notations such as route, depot, customer and vehicles. Fig. 9 shows the graphical model of a simple VRPTW and its solution. This example has two routes, R_1 and R_2, and every customer is given a number as its identity. The arrows connecting the customers show the sequences of visits by the vehicles, where every route must start and end at the depot.

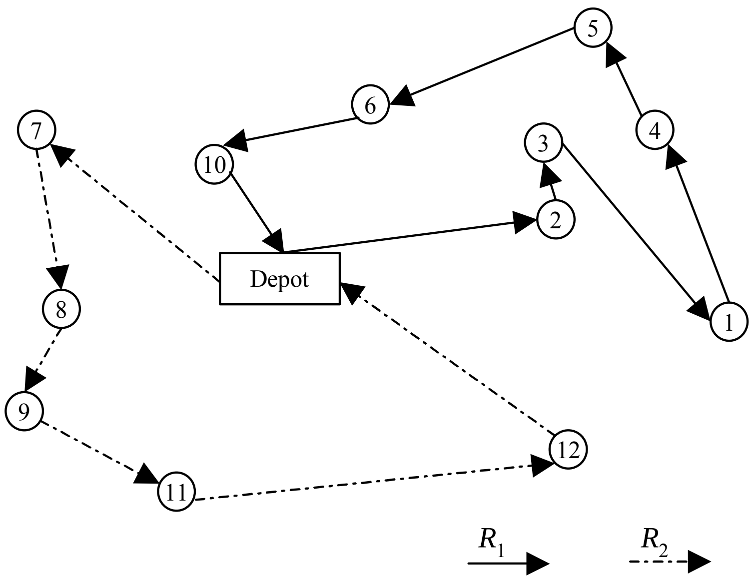


Figure 9 Graphical representation of a simple vehicle routing problem

The definition of the terms and constraints for the VRPTW is given as follows:

- *Depot*: The depot is denoted by v_0, which is a node where every vehicle must start and end its route. It does not have load but it has specified time window to be followed.

- *Customers*: There are N customers and the set $\{0, 1\ldots, N\}$ represents the sites of these N customers. The number 0 represents the depot. Every customer i has a demand, $k_i \geq 0$ and a service time, $s_i \geq 0$. Formally, $\Omega = \{0,1,2,...,N\}$ is the customer set and $\Omega(r)$ represents the set of customers served by a route r.

- *Vertex*: A vertex is denoted by $v_i(r)$, which represents the customer that is served at the i^{th} sequence in a particular route r. It must be an element in the customer set defined as $v_i(r) \in \Omega$.

- *Vehicles*: There are m identical vehicles and each vehicle has a capacity limit of K. The number of customers that a vehicle can serve is unlimited given that the total load does not exceed the capacity limit K. The vehicles may arrive before the earliest service time and thus may need to wait before servicing customers.

- *Traveling cost*: The traveling cost between customers i and j is denoted by c_{ij}, which satisfies the triangular inequality where $c_{ij} + c_{jk} \geq c_{ik}$. The cost is calculated with the following equation,

$$c_{ij} = \sqrt{(i_x - j_x)^2 + (i_y - j_y)^2} \qquad (1)$$

  where i_x is the coordinate x for customer i and i_y is the coordinate y for customer i. Clearly, the routing cost is calculated as Euclidian distance between the two customers. An important assumption is made here: one unit distance corresponds to one unit traveling time, i.e., every unit distance may take exactly a unit of time to travel. Therefore c_{ij} not only defines the traveling cost (distance) from customer *i* to customer *j*, but also specifies the traveling time from customer *i* to customer *j*.

- *Routes*: A vehicle's route starts at the depot, visits a number of customers, and returns to the depot. A route is commonly represented as $r = \langle v_0, v_1(r), v_2(r), ..., v_r(r), v_0 \rangle$. Since all vehicles must depart and return to the depot v_0, the depot can be omitted in the representation, i.e., $r = \langle v_1(r), v_2(r), ..., v_r(r) \rangle$. However, the costs from the depot to the first customer node and from the last customer node to the depot must be included in the computation of the total traveling cost.

- *Customers in a route*: The customers in a route are denoted by $\Omega(r) = \{v_1(r), ..., v_n(r)\}$. The size of a route, *n*, is the number of customers served by the route. Since every route must start and end at the depot implicitly, there is no need to include the depot in the notation of $\Omega(r)$.

- *Capacity*: The total demands served by a route, $k(r)$, is the sum of the demands of the customers in the route *r*, i.e., $k(r) = \sum_{i \in \Omega(r)} k_i$. A route satisfies its capacity constraint if $k(r) \le K$.

- *Traveling cost*: The traveling cost of a route $r = \langle v_1,...,v_n \rangle$, denoted by $t(r)$, is the cost of visiting all customers in the route, i.e., $t(r) = \sum_{i=1}^{n-1} \left(c_{v_i(r),v_{i+1}(r)} \right) + c_{v_0,v_1(r)} + c_{v_n(r),v_0}$

- *Routing plan*: The routing plan, G, consists of a set of routes $\{r_1,\ldots,r_m\}$. The number of routes should not exceed the maximum number of vehicles M allowed, i.e., $m \leq M$. The following condition that all customers must be routed and no customers can be routed more than once must be satisfied,

$$\begin{aligned} &\bigcup_{i=1}^{m} \Omega(r_i) = \Omega \\ &\Omega(r_i) \cap \Omega(r_j) = \varnothing, \quad i \neq j \end{aligned} \tag{2}$$

- *Time windows*: The customers and depot have time windows. The time window of a site, i, is specified by an interval $[e_{v_i(r)}, l_{v_i(r)}]$, where $e_{v_i(r)}$ and $l_{v_i(r)}$ represents the earliest and the latest arrival time, respectively. All vehicles must arrive at a site before the end of the time window $l_{v_i(r)}$. The vehicles may arrive earlier but must wait until the earliest time of $e_{v_i(r)}$ before serving any customers. The notation of e_{v_0} represents the time that all vehicles in the routing plan leave the depot, while l_{v_0} corresponds to the time that all vehicles must return to the depot. In fact, the interval [e_{v_0} , l_{v_0}] is the largest time window for which all customers' time windows must be within the range.

The earliest service time of vertex $v_i(r)$ is generally represented as $a_{v_i(r)}$ and the departure time from the vertex $v_i(r)$ is denoted by $d_{v_i(r)}$. The definitions of the earliest service time and the departure time are given as follows,

$$d_{v_0} = 0$$

$$a_{v_i(r)} = \max(d_{v_{i-1}(r)} + c_{v_{i-1}(r),v_i(r)}\ ,\ e_{v_i(r)}) \qquad \text{for } \forall r \text{ and } 1 \le i \le n$$

$$d_{v_i(r)} = a_{v_i(r)} + s_{v_i(r)} \qquad \text{for } \forall r \text{ and } 1 \le i \le n$$

$$d_{v_{n+1}(r)} = d_{v_n(r)} + c_{v_n(r),v_0} \qquad \text{for } \forall r$$

$d_{v_{n+1}(r)}$ is the completion time of a route or the time that a vehicle completes all its jobs.

where $v_{i\text{-}1}$ refers to information of the previous customer in a route. The time window constraints in the VRPTW model are given as,

$$d_{v_{n+1}(r)} \le l_{v_0} \qquad \text{for } \forall r \in G$$

$$\left.\begin{array}{l} a_{v_i(r)} \ge e_{v_i(r)} \\ a_{v_i(r)} \le l_{v_i(r)} \end{array}\right. \qquad \text{for } \forall r \in G \text{ and } 1 \le i \le n$$

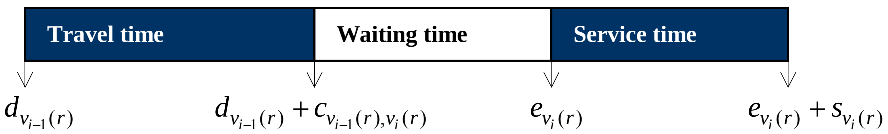
A solution to the VRPTW is a routing plan $G = \{r_1,\ldots,r_m\}$ satisfying both the capacity and time window constraints, i.e., for all routes,

$$k(r_j) \le K \qquad (3)$$

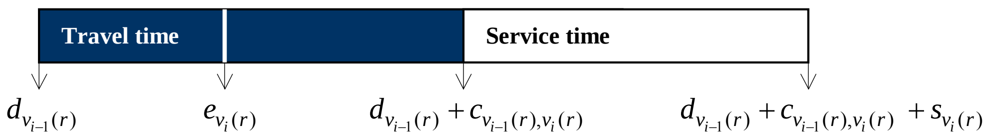
where $1 \leq j \leq m$. The VRPTW consists of finding a solution G that minimizes the number of vehicles and the total traveling cost as given below,

$$\begin{aligned} f(G)_1 &= |G| = m \\ f(G)_2 &= \sum_{r \in G} t(r) \end{aligned} \tag{4}$$

Both the capacity and time windows are specified as hard constraints in the VRPTW. As illustrated in Fig. 10, there are two possible scenarios based on the time window constraints in the model. As shown in Fig. 10(a), when a vehicle leaves the current customer and travels to the next customer, it may arrive before the earliest arrival time, $e_{v_i(r)}$, and therefore has to wait until the $e_{v_i(r)}$ starts. The vehicle will thus complete its service for this customer at the time of $e_{v_i(r)} + s_{v_i(r)}$. Fig. 10(b) shows the situation where a vehicle arrives at a customer node after the time window starts. In this case, the arrival time is $d_{v_{i-1}(r)} + c_{v_{i-1}(r),v_i(r)}$ and the vehicle will complete its service for customer i at the time of $d_{v_{i-1}(r)} + c_{v_{i-1}(r),v_i(r)} + s_{v_i(r)}$.



(a) Vehicle arrives before the earliest service time



(b) Vehicle arrives after the earliest service time

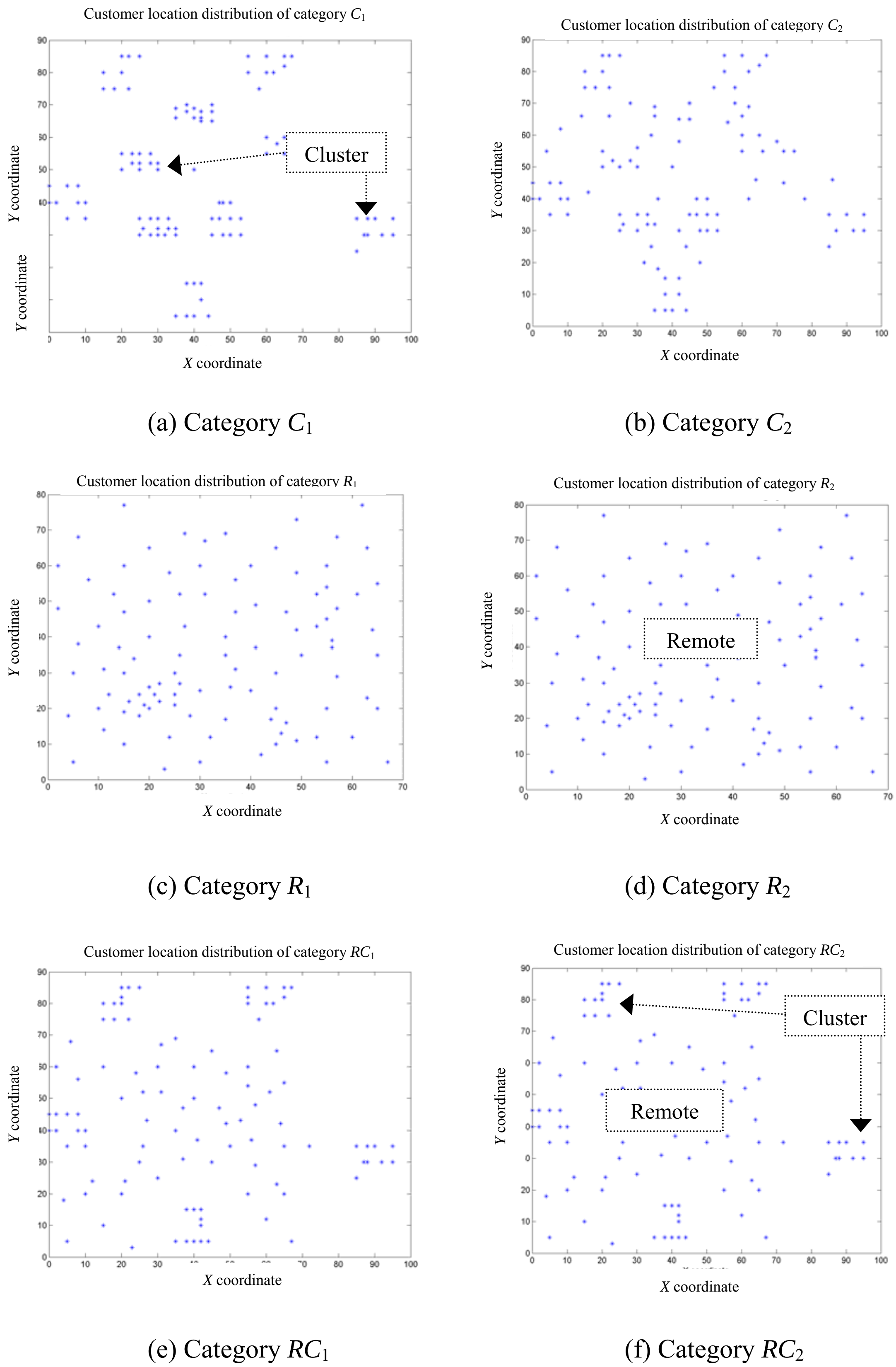
Figure 10 Examples of the time windows in VRPTW

4.2.2 The Solomon's 56 Benchmark Problems for VRPTW

The six benchmark problems (Solomon, 1987) designed specifically for the vehicle routing problem with time window constraints (VRPTW) are adopted in this research to illustrate the performance of the HMOEA. The Solomon's problems consist of 56 data sets, which have been extensively used for benchmarking different heuristics in literature over the years. The problems vary in fleet size, vehicle capacity, traveling time of vehicles, spatial and temporal distribution of customers. In addition to that, the time windows allocated for every customer and the percentage of customers with tight time-windows constraint also vary for different test cases. The customers' details are given in the sequence of customer index, location in *x* and *y* coordinates, the demand for load, the ready time, due date and the service time required. All the test problems consist of 100 customers, which are generally adopted as the problem size for performance comparisons in VRPTW. The traveling time between customers is equal to the corresponding Euclidean distance. The 56 problems are divided into 6 categories based on the pattern of customers' locations and time windows. These 6 categories are named as C_1, C_2, R_1, R_2, RC_1 and RC_2.

The problem category *R* has all customers located remotely and the problem category *C* refers to clustered type of customers. The *RC* is a category of problems having the mixed of remote and clustered customers. The geographical distribution determines the traveling distances between customers (Desrochers *et al*., 1992). In the cluster type of distribution, customers' locations are closer to each other and thus the traveling distances are shorter. In the remote type of distribution, customers' locations are remotely placed. Therefore the traveling distance is relatively longer in the *R* category as compared to the *C* category problems. Generally, the *C* category problems are easier to be solved because their solutions are less sensitive to the usually small distances among customers. In contrast, the *R* category problems require more efforts to obtain a correct sequence of customers in each route, and different sequences may result in large differences in term of the routing cost.

The data sets are further categorized according to the time windows constraints. The problems in category 1, e.g., C_1, R_1, RC_1, generally come with a smaller time window, and the problems in category 2, e.g., C_2, R_2 and RC_2 are often allocated with a longer time window. In the problem sets of R_1 and RC_1, the time windows are generated randomly. In the problem set of C_1, however, the variations of time windows are small. A shorter time window indicates that many candidate solutions can become infeasible easily after reproduction due to the tight constraint. In contrast, a larger time window means that more feasible solutions are possible and subsequently encourage the existence of longer routes, i.e., each vehicle can serve a larger number of customers. In Fig. 11, the *x*-*y* coordinate depicts the distribution of customers' locations for the six different categories, C_1, C_2, R_1, R_2, RC_1 and RC_2. Figs. 11(a), 11(c) and 11(e) are labeled with "cluster" or/and "remote" to show the distribution of customers corresponding to its problem category. For example, in Fig. 11(e), there exist two types of customer distribution patterns, i.e., cluster and remote, since the *RC* category consists of both the *R* and *C* type problems.



(a) Category C_1 (b) Category C_2

(c) Category R_1 (d) Category R_2

(e) Category RC_1 (f) Category RC_2

Figure 11 Customers' distribution for the problem categories of C_1, C_2, R_1, R_2, RC_1 and RC_2

4.3 A Hybrid Multiobjective Evolutionary Algorithm

The VRPTW can be best solved by means of multiobjective optimization, i.e., it involves optimizing routes for multiple vehicles to meet all constraints and to minimize multiple conflicting cost functions concurrently, such as the traveling distance and the number of vehicles. This section presents a hybrid multiobjective evolutionary algorithm specifically designed for the VRPTW. Section 4.3.1 gives a brief description of multiobjective evolutionary optimization and its applications in a number of domain-specific combinatorial optimization problems. The program flowchart of the HMOEA is illustrated in Section 4.3.2 to provide an overview of the algorithm. Sections 4.3.3-4.3.6 present the various features of HMOEA designed and incorporated to solve the multiobjective VRPTW optimization problem, including the variable-length chromosome representation in Section 4.3.3, specialized genetic operators in Section 4.3.4, and Pareto fitness ranking in Section 4.3.5. Following the concept of hybridizing local optimizers with multiobjective evolutionary algorithms as proposed by Tan *et al*., (2001c), Section 4.3.6 describes the various heuristics that are incorporated in HMOEA to improve its local search exploitation capability for VRPTW.

4.3.1 Multiobjective Evolutionary Optimization and Applications

Evolutionary algorithms (Bäck, 1996; Michalewicz *et al*., 1999) are global search optimization techniques based upon the mechanics of natural selection and reproduction, which have been found to be very effective in solving complex multiobjective optimization problems where conventional optimization tools fail to work well (Bagchi, 1999; Deb, 2001a; Fonseca and Fleming, 1993). Without the need of linearly combining multiple attributes into a composite scalar objective function, evolutionary algorithms incorporate the concept of Pareto's optimality to evolve a family of solutions at multiple points along the trade-off surface. Fig. 12 again shows a general Pareto dominance diagram with two solution points. Let f_1 and f_2 be two objectives in the VRPTW, a routing solution is Pareto-optimal if, in shifting from point *A* to another point *B* in the set, any improvement in one of the objective functions from its current value causes at least one of the other objective functions to deteriorate from its current value. Several surveys are available for more information of multiobjective evolutionary algorithms, e.g., Coello Coello (1999), Coello Coello *et al*., (2002), Fonseca (1995), Van Veldhuizen and Lamont (2000), and Zitzler and Thiele (1999).

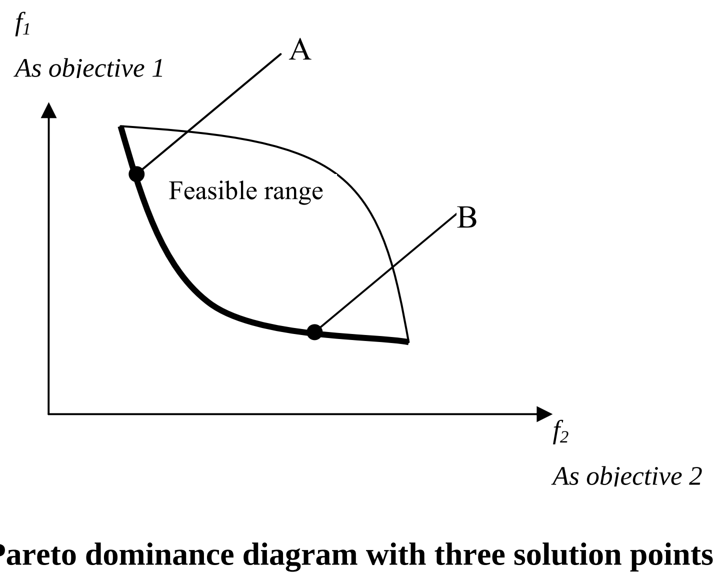


Figure 12 A Pareto dominance diagram with three solution points

Although multiobjective evolutionary algorithms have been applied to solve a number of domain-specific combinatorial optimization problems, such as flowshop/jobshop scheduling, nurse scheduling and timetabling, these algorithms are designed with specific representation or genetic operators that could only be used in particular application domains, and cannot be directly applied to solve the VRPTW addressed in this research. For example, Murata *et al*. (1996) presented two hybrid genetic algorithms (GAs) to solve a flowshop scheduling problem that is characterized by unidirectional flow of work with a variety of jobs being process sequentially in a one-pass manner. Jaszkiewicz (2001) proposed the algorithm of Pareto simulated annealing (PSA) to solve a multiobjective nurse scheduling problem. Chen *et al*. (1996) provided a GA-based approach to tackle continuous flowshop problem in which the intermediate storage is required for partially finished jobs. Dorndorf and Pesch (1995) proposed two different implementations of GA using priority-rule-based-representation and machine-based representation to solve a jobshop scheduling problem (JSP). The JSP concerns the processing on several machines with mutable sequence of operations, i.e., the flow of work may not be unidirectional as encountered in the flowshop problem. Ben *et al*. (1998) later devised a specific representation with two partitions in a chromosome to deal with the priority of events (in permutation) and to encode the list of possible time slots for events respectively. Jozefowiez *et al*. (2002) solved a multiobjective capacitated vehicle routing problem using a parallel genetic algorithm with hybrid Tabu search to increase the performance of the algorithm. Paquete and Fonseca, (2001) proposed an algorithm with modified mutation operator (and without recombination) to solve a multiobjective examination timetabling problem. It should be noted that although the methods described above shared a common objective of finding the optimal sequences in combinatorial problems, they are unique with different mathematical models, representations, genetic operators, and performance evaluation functions in their respective problem domains, which are different from that of the VRPTW problem.

4.3.2 Program Flowchart of HMOEA

Unlike many conventional optimization problems, the VRPTW does not have a clear neighborhood structure, i.e., it is difficult to trace or predict good solutions for VRPTW since feasible solutions may not be located at the neighborhood of any candidate solutions in the search space. The same observation can be found in many combinatorial optimization problems. To design an evolutionary algorithm that is capable of solving such a combinatorial and ordered-based multiobjective optimization problem, a few features such as variable-length chromosome representation, specialized genetic operators, Pareto fitness ranking, and efficient local search heuristics are incorporated in the HMOEA. The program flowchart of HMOEA is shown in Fig. 13. The simulation begins by reading in customers' data and constructing a list of customers' information. The pre-processing process builds a database for customers' information, including all relevant coordinates (position), customers' load, time windows, service times required and etc. An initial population is then built such that each individual must at least be a feasible candidate solution, i.e., every individual and route in the initial population must be feasible. The initialization process is random and starts by inserting customers one by one into an empty route in a random order. Any customer that violates any constraint is deleted from current route. The route is then accepted as part of the solution. A new empty route is added to serve the deleted customer and the other remaining customers. This process continues until all customers are routed and a feasible initial population is built as depicted in Fig. 14.

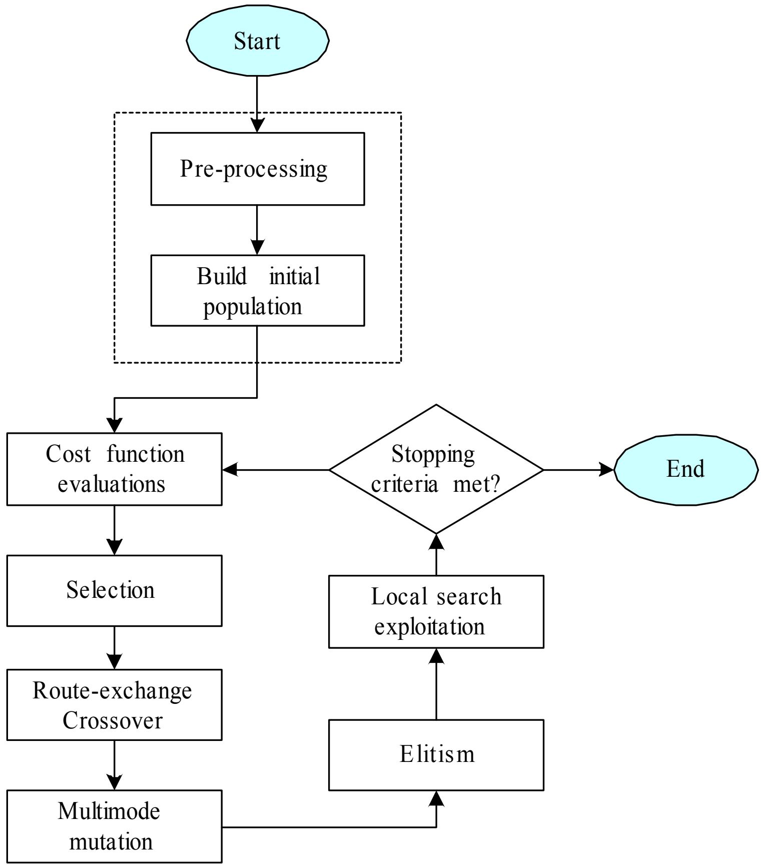


Figure 13 The program flowchart of HMOEA

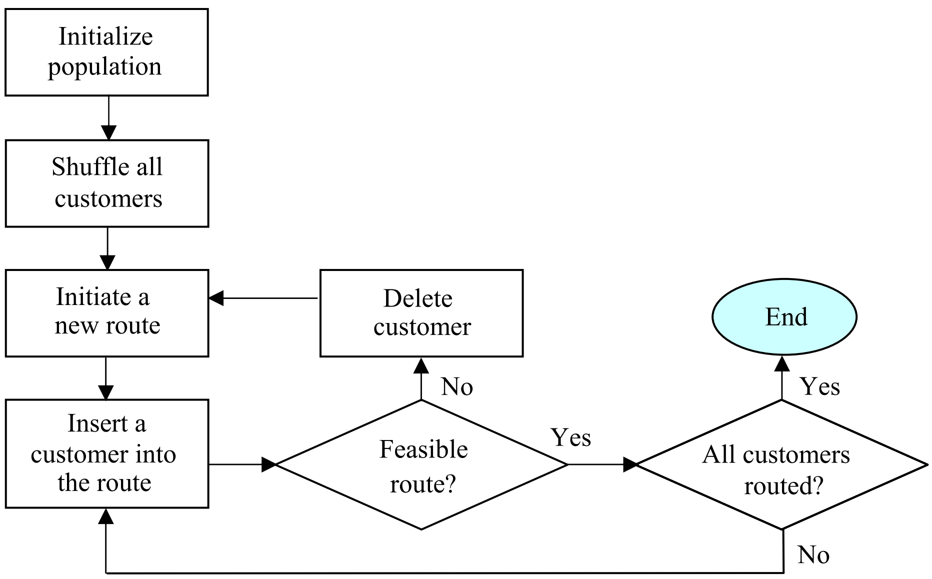


Figure 14 The procedure of building an initial population of HMOEA

After the initial population is formed, all individuals will be evaluated based on the objective functions as given in equation (4) and ranked according to their respective Pareto's dominance in the population. After the ranking process, tournament selection scheme (Tan *et al.*, 2001c) with a tournament size of 2 is performed, where all individuals in the population are randomly grouped into pairs and those individuals with a lower rank will be selected for reproduction. The procedure is performed twice to preserve the original population size. A simple elitism mechanism (Tan *et al.*, 2001c) is employed in the HMOEA for faster convergence and better routing solutions. The elitism strategy keeps a small number of good individuals (0.5% of the population size) and replaces the worst individuals in the next generation, without going through the usual genetic operations. The specialized genetic operators in HMOEA consist of route-exchange crossover and multimode mutation. To further improve the internal routings of individuals, heuristic searches are incorporated in the HMOEA at every 50 generations (after considering the trade-off between optimization performance and simulation time) for better local exploitation in the evolutionary search. It should be noted that the feasibility of all new individuals reproduced after the process of specialized genetic operations and local search heuristics is retained, which avoids the need of any repairing mechanisms. The evolution process repeats until the stopping criterion is met or no significant performance improvement is observed over the last 10 generations.

4.3.3 Variable-Length Chromosome Representation

The chromosomes in evolutionary algorithms, such as genetic algorithms, are often represented as a fixed-structure bit string, for which the bit positions are assumed to be independent and context insensitive. Such a representation is not suitable for VRPTW, which is an order-oriented NP-hard optimization problem where sequences among customers are essential. In HMOEA, a variable-length chromosome representation is applied such that each chromosome encodes a complete solution including the number of routes/vehicles and the customers served by these vehicles. Depending on how the customers are routed and distributed, every chromosome can have different number of routes for the same data set. As shown in Fig. 15, a chromosome may consist of several routes and each route or gene is not a constant but a sequence of customers to be served. Such a variable-length representation is efficient and allows the number of vehicles to be manipulated and minimized directly for multiobjective optimization in VRPTW. It should be noted that most existing routing approaches only consider a single objective/cost of traveling distance, since the number of vehicles is often incontrollable in these representations.

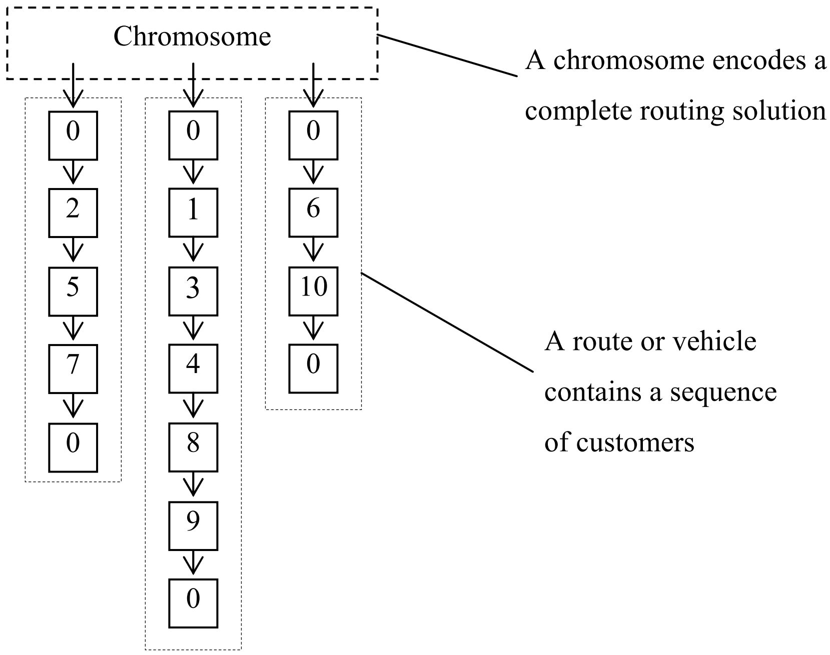


Figure 15 The data structure of the chromosome representation in HMOEA

4.3.4 Specialized Genetic Operators

Since standard genetic operators may generate individuals with infeasible routing solutions for VRPTW, the specialized genetic operators of route-exchange crossover and multimode mutation are incorporated in the HMOEA, which are described in the following sub-sections.

4.3.4.1 Route-exchange Crossover

Classical one-point crossover may produce infeasible route sequence because of the duplication and omission of vertices after reproduction. Goldberg and Lingle (1985) proposed a PMX crossover operator suitable for sequencing optimization problem. The operator cuts out a section of the chromosome and puts it in the offspring. It maps the remaining sites to the same absolute position or the corresponding bit in the mate's absolute position to avoid any redundancy. Whitley *et al.* (1989) proposed a genetic edge recombination operator to solve a TSP problem. For each node, an edge-list containing all nodes is created. The crossover parents shared the edge-lists where several manipulations on edge-list are repeated until all edge-lists are processed. Ishibashi *et al.* (2000) proposed a two-point ordered crossover that randomly selects two crossing points from parents and decides which segment should be inherited to the offspring.

This research proposes a simple crossover operator for HMOEA that allows the good sequence of routes or genes in a chromosome to be shared with other individuals in the population. The operation is designed such that infeasibility after the change can be eradicated easily. The good routes in VRPTW are those with customers/nodes arranged in sequence where the cost of routing (distance) is small and the time window fits perfectly one after another. In a crossover operation, the chromosomes would share their best route to each other as shown in Fig. 16. The best route is chosen according to the criteria of averaged cost over nodes, which can be computed easily based on the variable-length chromosome representation in HMOEA. To ensure the feasibility of chromosomes after the crossover, each customer can only appear once in a chromosome, i.e., any customer in a chromosome will be deleted during the insertion of new routes if the customer is also found in the newly inserted route. The crossover operation will not cause any violation in time windows or capacity constraints. Deleting a customer from a route will only incur some waiting time before the next customer is serviced, and thus will not cause any conflicts for the time windows. Meanwhile, the total load in a route will only be decreased when a customer is deleted from the route, and thus will not violate any capacity constraints. Therefore all chromosomes will remain feasible routing solutions after the crossover in HMOEA.

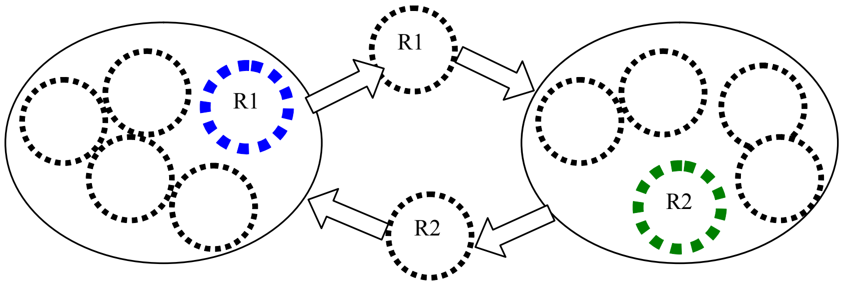


Figure 16 The route-exchange crossover in HMOEA

4.3.4.2 Multimode Mutation

Gendreau *et al*. (1999) proposed a RAR (remove and reinsert) mutation operator, which extracts a node and inserts it into a random point of the routing sequence in order to retain the feasibility of solutions. Ishibashi *et al*. (2000) extends the approach to a shift mutation operator that extracts a segment or a number of nodes (instead of a node) and inserts it into a new random point for generating the offspring. During the crossover by HMOEA, routes' sequence is exchanged in a whole chunk and no direct manipulation is made to the internal ordering of the nodes for the VRPTW. The sequence in a route is modified only when any redundant nodes in the chromosome are deleted. In this research, a multimode mutation operator is proposed in the HMOEA, which serves to complement the crossover by optimizing the local route information of a chromosome. As shown in Fig. 17, there are three parameters related to the multimode mutation, i.e., mutation rate (*PM*), elastic rate (*PE*) and squeeze rate (*PS*). In HMOEA, random numbers will be generated and compared to these parameter values in order to determine if the mutation operations (for each mutation type) should be performed.

The mutation rate is considerably small since it could be destructive to the chromosome structure and information of routes. In order to trigger more moves with better routing solutions, a few operations including Partial_Swap (Bagchi, 1999), Split_Route and Merge_Routes (Pinaki and Elizabeth, 1999) are implemented. In this case, only one operation is chosen if mutation happens. The elastic rate determines the operation of Partial_Swap, which picks two routes in a chromosome and swaps the two routes at a random point that has a value smaller or equal to the shortest size of the two chosen routes. The swapping must be feasible or else the original routes will be restored. The squeeze rate determines the operation of splitting or merging a route. The Split_Route operation breaks a route at a random point and generates two new feasible routes. The operation has an always-true condition, unless the number of vehicles exceeds the maximum vehicles allowed. A number of constraints should be satisfied in the operation of Merge_Routes, e.g., it should avoid any violation against the hard constraints, such as time windows and vehicle capacity. During the Merge_Routes operation, the two routes with the smallest number of customers are chosen, and these routes must have the capacity to accommodate additional customers. Let the two selected routes be route *A* and route *B*, the operation first inserts all customers, one by one, from route *B* into route *A*. If there is any violation against the capacity or time window constraints in route *A*, the remaining nodes will be kept at the route *B*. If all the customers in route *B* are shifted to route *A*, then the route *B* will be deleted.

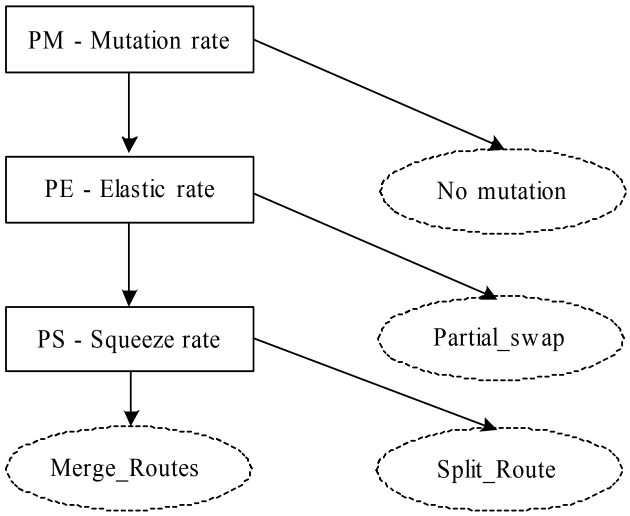


Figure 17 The multimode mutation in HMOEA

4.3.5 Pareto Fitness Ranking

The VRPTW is a multiobjective optimization problem where a number of objectives such as the number of vehicles (*NV*) and the cost of routing (*CR*) as given in eqn. 4 need to be minimized concurrently, subject to some constraints like time window and vehicle capacity. Fig. 18 illustrates the concept of multiobjective optimization in VRPTW, for which the small boxes represent the solutions resulted from an optimization. Point '*d*' is the minimum solution for both the objectives of *NV* and *CR*, which is sometimes infeasible or cannot be obtained. Point '*b*' is a compromised solution between the cost of routing (*CR*) and the number of vehicles (*NV*). If a single-objective routing method is employed, its effort to push the solution towards point '*b*' may lead to the solution of point '*a*' (if only the criterion of *CR* is considered) or the solution of point '*c*' (if only the criterion of *NV* is considered). Instead of giving only a particular solution, the HMOEA for multiobjective optimization in VRPTW aims to discover the set of non-dominated solutions concurrently, i.e., points '*a*', '*b*' and '*c*' together, for which the designer could select an optimal solution depending on the current situation, as desired.

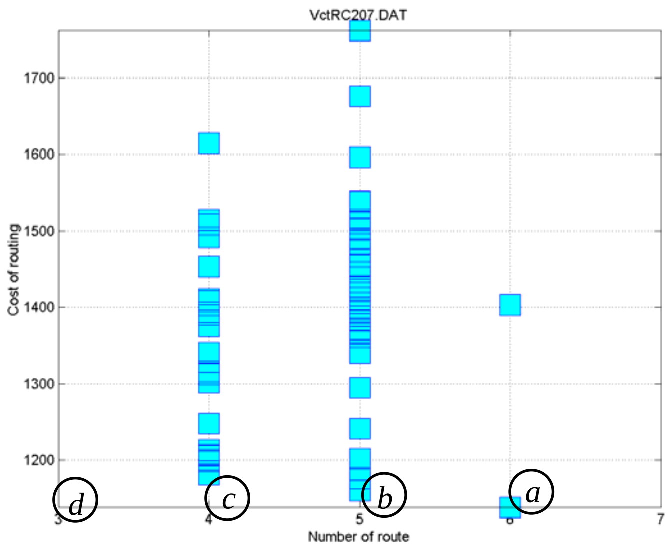


Figure 18 Trade-off graph for the cost of routing and the number of vehicles

The Pareto fitness ranking scheme (Fonseca, 1995; Tan *et al*., 2001c; Fonseca and Fleming, 1998) for evolutionary multiobjective optimization is adopted here to assign the relative strength of individuals in the population. The ranking approach assigns the same smallest rank for all non-dominated individuals, while the dominated individuals are inversely ranked according to how many individuals in the population dominating them based on the criteria below:

- A smaller number of vehicles but an equal cost of routing

- A smaller routing cost but an equal number of vehicles
- A smaller routing cost and a smaller number of vehicles

Therefore the rank of an individual p in a population is given by $(1+q)$, where q is the number of individuals that dominate the individual p based on the above criteria.

4.3.6 Local Search Exploitation

As stated by Tan *et al*., (2001c), the role of local search is vital in multiobjective evolutionary optimization in order to encourage better convergence and to discover any missing trade-off regions. The local search approach can contribute to the intensification of the optimization results, which is usually regarded as a complement to evolutionary operators that mainly focus on global exploration. Jaszkiewicz (1998) proposed a multiobjective metaheuristic based on the approach of local search to generate a set of solutions approximate to the whole non-dominated set of a traveling salesman problem. For the problem of VRPTW as addressed in this research, the local search exploitation is particularly useful for solving the problem of R category, where the customers are far away from each other and the swapping of 2 nodes in a route implemented by the local optimizers could improve the cost of routing significantly. Three famous local heuristics are incorporated in the HMOEA to search for better routing solutions in the VRPTW, which include the Intra_Route, Lambda_Interchange (Osman and Christofides, 1989), and Shortest_pf (Lin, 1965). Descriptions of these heuristics are given in Table 5. There is no preference made among the local heuristics, and one of them will be randomly executed at the end of every 50 generations for all individuals in a population to search for better local routing solutions.

Table 5 The three local heuristics incorporated in HMOEA

Local heuristic	Description
Intra_Route	This heuristic picks two routes randomly and swaps two nodes from each route. The nodes are chosen based on the numbers generated randomly. After the swapping is done, feasibility is checked for the newly generated routes. If the two new routes are acceptable, they will be updated as part of the solutions; otherwise the original routes will be restored.
Lambda_Interchange	This heuristic is cost-oriented where a number of nodes will be moved from one route into another route. Assume two routes *A* and *B* are chosen; the heuristic starts by scanning through nodes in route *A* and moves the feasible node into route *B*. The procedure repeats until a pre-defined number of nodes are shifted or the scanning ends at the last node of route *A*.
Shortest_pf	This heuristic is modified from the 'shortest path first' method. It attempts to rearrange the order of nodes in a particular route such that the node with the shortest distance is given priority. For example, given a route *A* that contains 5 customers, the first node is chosen based on its distance from the depot and the second node is chosen based on its distance from the first customer node. The process repeats until all nodes in the original route are re-routed. The original route will be restored if the new route obtained is infeasible.

4.4 Simulation Results and Comparisons

Section 4.4.1 presents the system specification of the HMOEA and the detail setup of the experiments. The advantages of HMOEA for multiobjective optimization in VRPTW, such as lower routing cost, wider scattering area and better convergence trace as compared with conventional single-objective approaches are described in Section 4.4.2. Section 4.4.3 includes some performance comparisons for the features incorporated in HMOEA such as the proposed genetic operators and the local search heuristics. Section 4.4.4 presents the extensive simulation results of HMOEA based upon the famous Solomon's 56 data sets where statistical significance of the results was studied as well. The performance of the HMOEA is compared with the best-known VRPTW results published in literature.

4.4.1 System Specification and Experiment Setup

The HMOEA was programmed in C++ based on a Pentium III 933 MHz processor with 256 MB RAM under the Microsoft Windows 2000 operating system. The vehicle, customer, route sequence and set of solutions are modeled as classes of objects. The class of node is the fundamental information unit concerning a customer. The class of route is a vector of nodes, which describes a continuous sequence of customers by a particular vehicle. The class of chromosome consists of a number of routes that carries the solution of the routing problem. Constraints and objectives are modeled as behaviors in the classes, e.g., a predefined number limits the maximum capacity of a vehicle which is included as one of the behaviors in the route. In all simulations, the following parameter settings were chosen after some preliminary observations:

Crossover rate	= 0.7
Mutation rate	= 0.1
Elastic rate	= 0.5
Squeeze rate	= 0.7
Elitism rate	= 0.5% of the population size
Population size	= 1000
Generation size	= 1500 or no improvement over the last 10 generations

4.4.2 Multiobjective Optimization Performance

This section presents the routing performances of HMOEA, particularly on its multiobjective optimization that offer the advantages of improved routing solutions, wider scattering area and better convergence trace over conventional single-objective routing approaches.

In vehicle routing problems, the logistic manager is often not only interested in getting the minimum routing cost, but also the smallest number of vehicles required to service the plan. Ironically, in many literatures especially the classical models are often formulated and solved with respect to a particular cost or by linearly combining the multiple objectives into a scalar objective via a predetermined aggregating function to reflect the search for a particular solution. The drawback of such an objective reduction approach is that the weights are difficult to determine precisely, particularly when there is often insufficient information or knowledge concerning the large real-world vehicle routing problem. Clearly, these issues could be easily addressed via the proposed HMOEA that optimizes both objectives concurrently and effectively without the need of any calibration of weighting coefficients.

In the VRPTW model as formulated in Section 4.2, there are two objectives including the number of vehicles and the total traveling cost need to be optimized concurrently. Although both the objectives are quantitatively measurable, the relationship between these two values in a routing problem is unknown until the problem has been solved. These two objectives may be positively correlated with each other, or they may be conflicting to each other. For example, fewer vehicles employed in service do not necessarily increase the routing cost. On the other hand, higher routing cost may be incurred if more vehicles are involved. From the computational results of the Solomon's 56 data sets, an analysis is carried out to count the number of problem instances with conflicting objectives as well as the number of instances having positively correlating objectives. As shown in Fig. 19, although all instances in the categories of C_1 and C_2 are having positively correlating objectives (the routing cost of a solution is increased as the number of vehicles is increased), there are many instances in R_1, R_2, RC_1 and RC_2 categories that are having conflicting objectives (the routing cost of a solution is reduced as the number of vehicles is increased). Obviously, such a relationship (conflicting or positively correlating) between the two objectives in a routing problem could be easily discovered using the proposed HMOEA, but is hard to be found if conventional single-objective vehicle routing approaches are used.

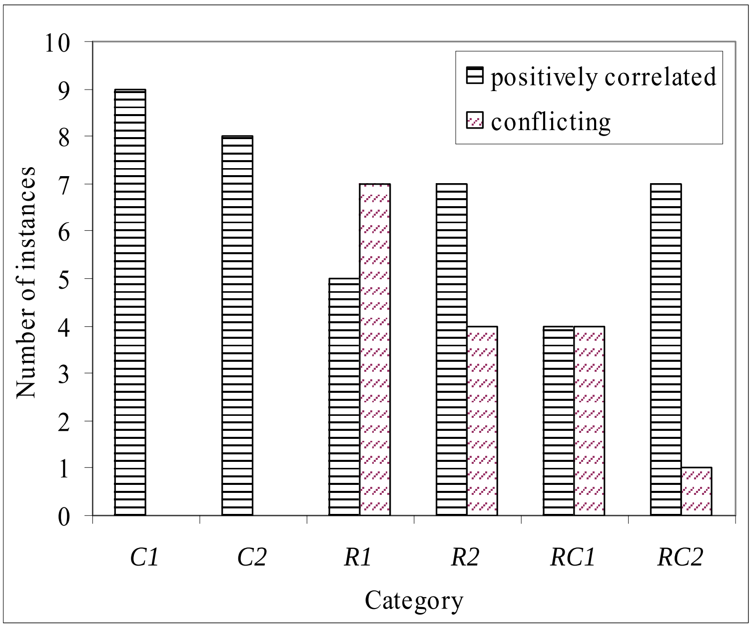


Figure 19 Number of instances with conflicting and positively correlating objectives

To illustrate the performance of HMOEA, three types of simulations with similar settings but different set of optimization criteria (for evolutionary selection operation) in VRPTW have been performed, i.e., each type of simulation concerns the optimization criterion of routing cost (*CR*), vehicle numbers (*NV*), and multiple objectives (*MO*) including *CR* and *NV*, respectively. Fig. 20 shows the comparison results for the evolutionary optimization based upon the criterion of *CR*, *NV*, and *MO*, respectively. The comparison was performed using the multiplicative aggregation method (Van Veldhuizen, 1998) of average cost and average number of routes for the different categories of data sets. The results of C_1 category is omitted in the figure since no significant performance difference is observed for this data set.

As can be seen, the *MO* produces the best performance with the smallest value of *CR*×*NV* for all the categories.

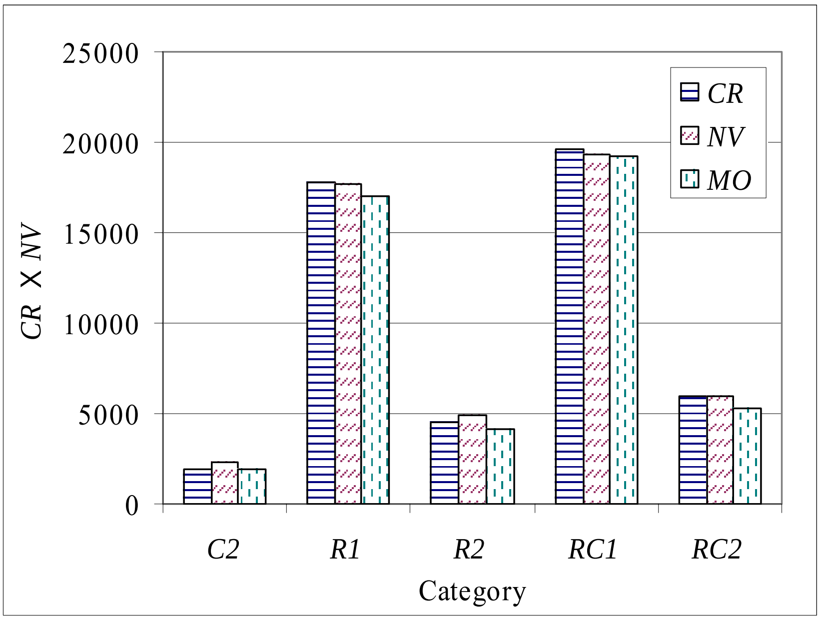


Figure 20 Performance comparisons for different optimization criteria of *CR*, *NV* and *MO*

In general, multiobjective optimization tends to evolve a family of points that are widely distributed or scattering in the objective domain such that a broader coverage of solutions is possible. Fig. 21 illustrates the distribution of individuals in the objective domain (*CR vs. NV*) for one randomly selected instance in each of the five categories of data sets. In the figure, each individual in a population is plotted as a small box based on its performance of *CR* and *NV*. A portion appears darker than others when its solution points are congested in the graph. In contrast, a portion looks lighter if its solution points are fairly distributed in the objective domain. As can be seen, all graphs in Fig. 21 using the optimization criteria of *MO* appear to be fairly distributed over a large area in the objective domain. This can also be illustrated from the measure of scattering points by dividing the entire interested region in the objective domain into grids. If any individual exists in a grid, one scattering point is counted regardless of the number of individuals in that particular grid. Table 6 shows the percentage of area covered by scattering points. As shown in the table, *MO* outperforms the *CR* and *NV* by scoring the highest percentage for all the 5 categories of data sets. For example, in category $RC_{1\text{-}07}$, *MO* scored 40.00% area while *CR* and *NV* scored only 24.00% and 22.67%, respectively.

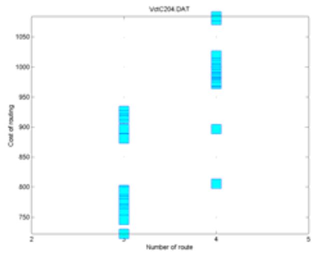
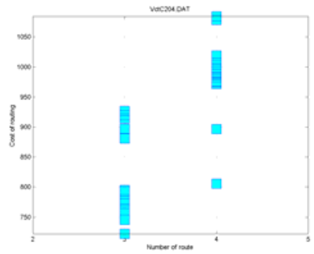
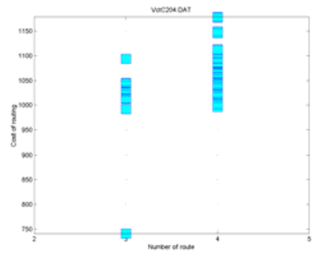
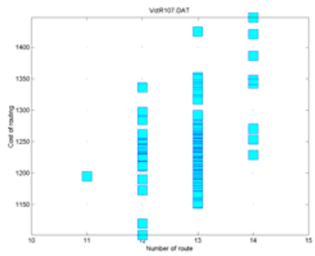
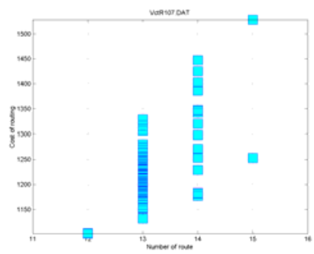
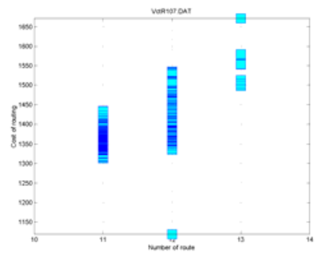
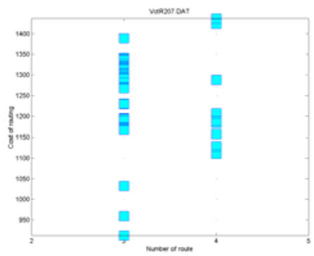
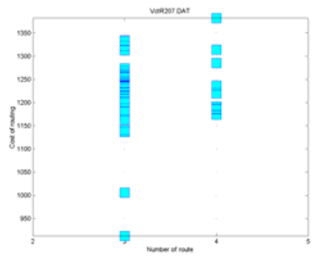
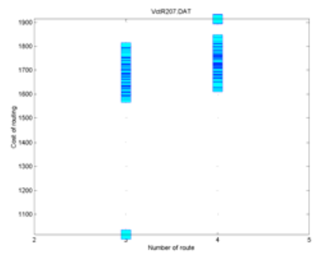
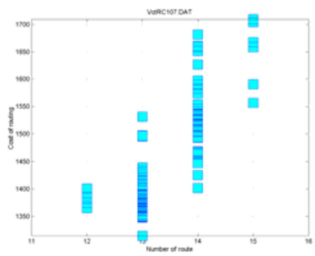
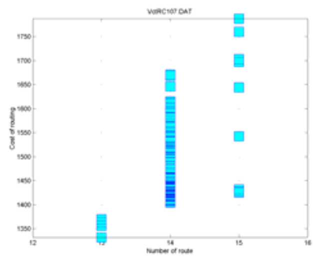
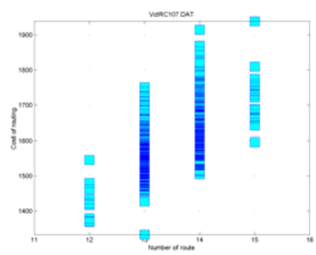
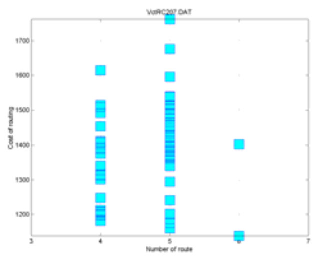
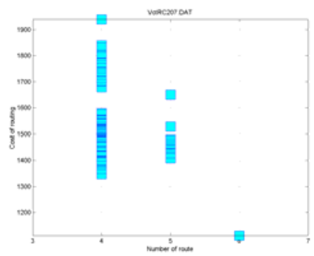
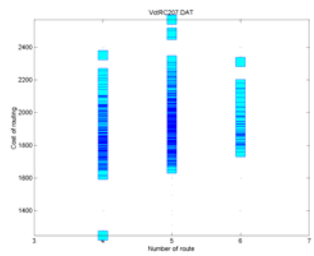
Cat.	*MO*	*CR*	*NV*
*C*2-04			
*R*1-07			
*R*2-07			
*RC*1-07			
*RC*2-07			

Figure 21 Comparison of population distribution for *CR*, *NV* and *MO*

Table 6 Comparison of scattering points for *CR*, *NV* and *MO*

Category	Objective space covered by scattering points (%)		
	CR	*NV*	*MO*
$C_{2\text{-}04}$	17.00	16.00	23.00
$R_{1\text{-}07}$	19.05	15.71	25.71
$R_{2\text{-}07}$	11.11	8.89	12.22
$RC_{1\text{-}07}$	24.00	22.67	40.00
$RC_{2\text{-}07}$	14.76	20.00	23.33

4.4.3 Specialized operators and Hybrid Local Search Performance

In this section, the performance in HMOEA is compared with two variants of evolutionary algorithms, i.e., MOEA with standard genetic operators as well as MOEA without hybridization of local search. The comparison allows the effectiveness of the various features in HMOEA, such as the specialized genetic operators and local search heuristic, to be examined.

A. Specialized genetic operators

In this experiment, two genetic operators commonly found in the literatures are devised to solve the VRPTW. The multiobjective evolutionary algorithm with standard generic operators (STD_MOEA) devised the commonly-known cycle crossover and RAR mutation. The cycle crossover is a general crossover operator that preserves the order of sequence in the parent partially and was applied to solve the traveling salesman problems by Oliver *et al*. (1987). The remove and reinsert (RAR) mutation operator removes a task from the sequence and reinsert it to a random position (Gendreau *et al*., 1999). The experiment setups and parameters for

STD_MOEA are similar to the settings for HMOEA (except that Elastic rate and Squeeze rate are not required in the RAR mutation operator). The specialized operators in HMOEA work efficiently for the purpose of multiobjective optimization especially for this vehicle routing problem as the representation is unique.

Fig. 22 shows the average values for the two objectives in VRPTW for all the 56 results. As shown in the figure, the STD_MOEA (the lines with larger markers) tend to incur higher cost and higher number of vehicles. The specialized operators in HMOEA have performed better in overall with lower objective values. The HMOEA's operators exploit some important information from the problem domain. The preservation of feasible routes to next generation is easier when using the specialized operators as to compare the common genetic operators that do not exploit the knowledge from problem representation. Since the search space of the multiobjective VRPTW optimization is complex, it is expected that the problem-specific HMOEA should provide an efficient and high-performance routing solution for such a problem, as illustrated by the simulation results.

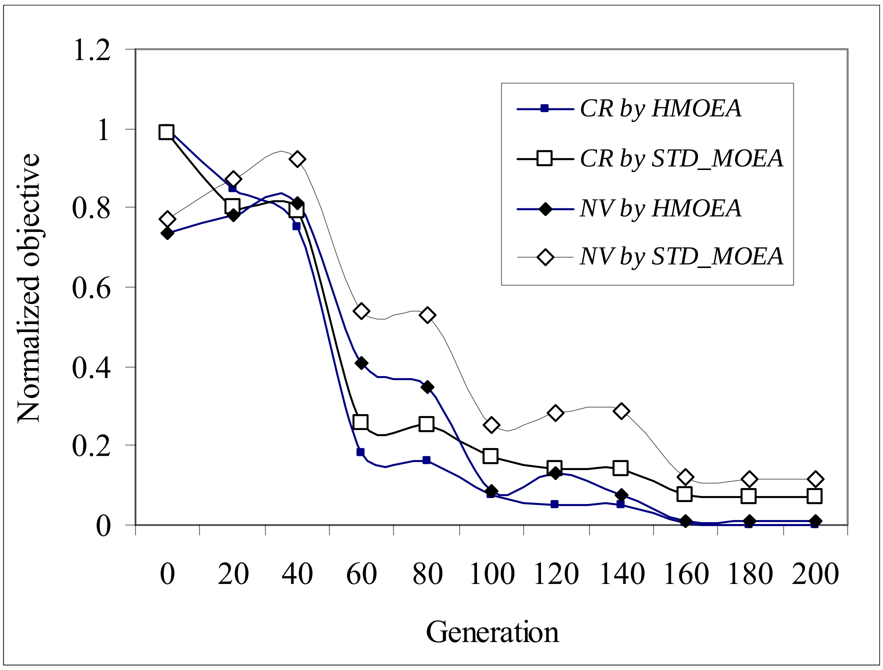
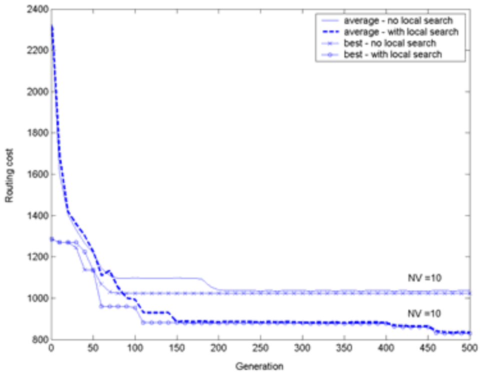


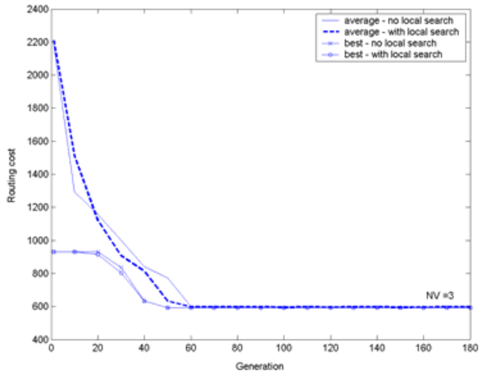
Figure 22 Comparison of performance for different genetic operators

B. Hybrid Local Search Performance

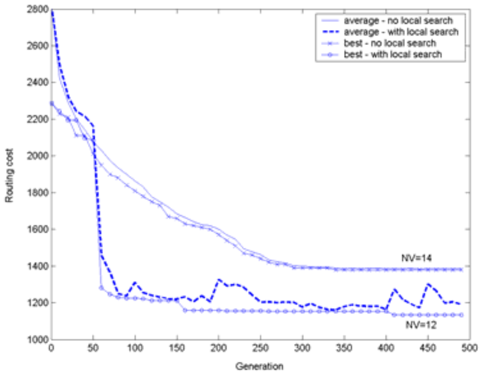
The HMOEA incorporates the local search heuristics in order to exploit local routing solutions in parallel with global evolutionary optimization. To demonstrate the effectiveness of local exploitation in HMOEA, the convergence trace of the best and average routing costs in a population for six randomly selected instances (one from each category) with and without the local search are plotted in Fig. 23. In the figure, the *NV* indicates the number of vehicles needed for the convergence with the best routing cost in the instances. As shown in Fig. 23, the HMOEA hybrid with local search performs better by having lower routing costs (*CR*) and smaller number of vehicles (*NV*) for almost all instances than the one without any local exploitation. It has also been observed that other instances in the Solomon's 56 data sets exhibit similar convergence performances as those shown in Fig. 23, which confirm the importance of incorporating local search exploitation in HMOEA.



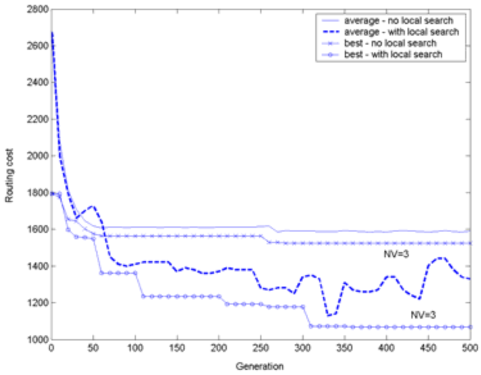
(a) $C_{1\text{-}09}$



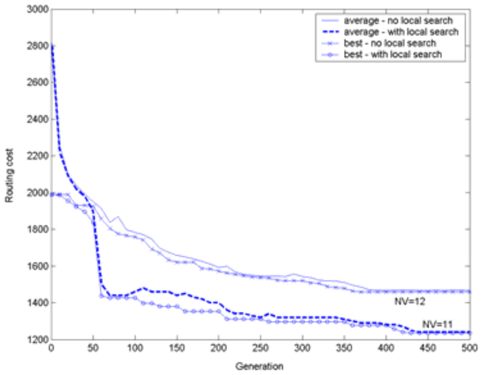
(b) $C_{2\text{-}01}$



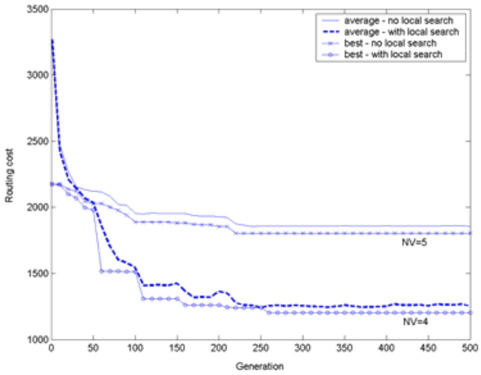
(c) $R_{1\text{-}07}$



(d) $R_{2\text{-}07}$



(e) $RC_{1\text{-}07}$



(f) $RC_{2\text{-}07}$

Figure 23 Comparison of simulations with and without local search exploitation in HMOEA

4.4.4 Performance Comparisons

In this section, the results obtained from HMOEA are compared with the best-known routing solutions obtained from different heuristics published in the literature according to the authors' best knowledge. Table 7 shows the comparison results between HMOEA and the best-known results in literature, for which instances with significant results or improvements are bolded. The solutions were selected from the results of optimization using HMOEA based upon the routing cost (CR). If the CR is similar, then the number of routes is considered. This is because the routing cost has been the benchmark used to compare the performances in traditional single objective optimization approaches. However, it is important to reiterate that no preference has been defined between the two objectives when solving the problem from multiobjective optimization approach. It can be seen that HMOEA produces excellent routing results with 20 data sets (out of the Solomon's 56 data sets) achieving a lower routing cost as compared to the best-known solutions obtained from various heuristics over the years. Besides, HMOEA also gives competitive routing solutions for 18 instances with similar or smaller number of vehicles and slightly higher routing cost (1%-2% in average) as compared to the best-known VRPTW solutions in literature.

Table 7 Comparison results between HMOEA and the best-known routing solutions

Data Set	**Best-Known Result**		**Source***	**HMOEA**	
	NV	*CR*		*NV*	*CR*
$C_{1\text{-}01}$	10	827.3	Desrochers *et al.*, (1992)	10	828.93
$C_{1\text{-}02}$	10	827.3	Desrochers *et al.*, (1992)	10	828.19

$C_{1\text{-}03}$	10	826.3	Taveres *et al.*, (2003)	10	828.06
$C_{1\text{-}04}$	10	822.9	Taveres *et al.*, (2003)	10	825.54
$C_{1\text{-}05}$	10	827.3	Taveres *et al.*,(2003)	10	828.90
$C_{1\text{-}06}$	10	827.3	Desrochers *et al.*, (1992)	10	828.17
$C_{1\text{-}07}$	10	827.3	Taveres *et al.*, (2003)	10	829.34
$C_{1\text{-}08}$	10	827.3	Taveres *et al.*, (2003)	10	832.28
$C_{1\text{-}09}$	10	827.3	Taveres *et al.*, (2003)	10	829.22
$C_{2\text{-}01}$	3	589.1	Cook and Rich, (1999)	3	591.58
$C_{2\text{-}02}$	3	589.1	Cook and Rich, (1999)	3	591.56
$C_{2\text{-}03}$	3	591.17	Li and Lim, (2002)	3	593.25
$C_{2\text{-}04}$	3	590.6	Potvin and Bengio, (1996)	3	595.55
$C_{2\text{-}05}$	3	588.88	De Backer *et al.*, (2002)	**3**	**588.16**
$C_{2\text{-}06}$	3	588.49	Lau *et al.*, (2001)	3	588.49
$C_{2\text{-}07}$	3	588.29	Rochat and Tailard, (1995)	3	588.88
$C_{2\text{-}08}$	3	588.32	Rochat and Tailard, (1995)	**3**	**588.03**
$R_{1\text{-}01}$	18	1607.7	Desrochers *et al.*, (1992)	18	1613.59
$R_{1\text{-}02}$	17	1434	Desrochers *et al.*, (1992)	18	1454.68
$R_{1\text{-}03}$	13	1175.67	Lau *et al.*, (2001)	14	1235.68
$R_{1\text{-}04}$	10	982.01	Rochat and Tailard, (1995)	**10**	**974.24**
$R_{1\text{-}05}$	15	1346.12	Kallehauge *et al.*, (2001)	15	1375.23
$R_{1\text{-}06}$	13	1234.6	Cook and Rich, (1999)	13	1260.20
$R_{1\text{-}07}$	11	1051.84	Kallehauge *et al.*, (2001)	11	1085.75
$R_{1\text{-}08}$	9	960.88	Berger *et al.*, (2001)	**10**	**954.03**
$R_{1\text{-}09}$	12	1013.2	Chiang and Russel, (1997)	12	1157.74
$R_{1\text{-}10}$	12	1068	Cook and Rich, (1999)	12	1104.56
$R_{1\text{-}11}$	12	1048.7	Cook and Rich, (1999)	12	1057.80
$R_{1\text{-}12}$	10	953.63	Rochat and Tailard, (1995)	10	974.73
$R_{2\text{-}01}$	4	1252.37	Homberger and Gehring, (1999)	**5**	**1206.42**
$R_{2\text{-}02}$	3	1158.98	Lau *et al.*, (2003)	**4**	**1091.21**
$R_{2\text{-}03}$	3	939.50	Lim and Zhang, (2005)	**4**	**935.04**
$R_{2\text{-}04}$	2	825.52	Bent and Van, (2001)	**3**	**789.72**
$R_{2\text{-}05}$	3	994.42	Rousseau *et al.*, (2002)	3	1094.65
$R_{2\text{-}06}$	3	833	Thangiah *et al.*, (1994)	3	940.12
$R_{2\text{-}07}$	3	814.78	Rochat and Tailard, (1995)	3	852.62
$R_{2\text{-}08}$	2	731.23	Homberger and Gehring, (1999)	2	790.60
$R_{2\text{-}09}$	3	855	Thangiah *et al.*, (1994)	3	974.88
$R_{2\text{-}10}$	3	954.12	Berger *et al.*, (2001)	5	982.31

$R_{2\text{-}11}$	2	892.71	Bent and Van, (2001)	**4**	**811.59**
$RC_{1\text{-}01}$	15	1619.8	Kohl *et al.*, (1999)	16	1641.65
$RC_{1\text{-}02}$	13	1530.86	Cordone and Wolfler, (2001)	**13**	**1470.26**
$RC_{1\text{-}03}$	11	1261.67	Shaw, (1998)	11	1267.86
$RC_{1\text{-}04}$	10	1135.48	Cordeau *et al.*, (2001)	10	1145.49
$RC_{1\text{-}05}$	13	1629.44	Lim and Zhang, (2005)	**14**	**1589.91**
$RC_{1\text{-}06}$	12	1395.4	Chiang and Russel, (1997)	**13**	**1371.69**
$RC_{1\text{-}07}$	11	1230.5	Taillard *et al.*, (1997)	**11**	**1222.16**
$RC_{1\text{-}08}$	10	1139.8	Taillard *et al.*, (1997)	**11**	**1133.90**
$RC_{2\text{-}01}$	4	1249	Thangiah *et al.*, (1994)	**6**	**1134.91**
$RC_{2\text{-}02}$	4	1164.3	Taillard *et al.*, (1997)	**5**	**1130.53**
$RC_{2\text{-}03}$	3	1049.62	Czech and Czarnas, (2002)	**4**	**1026.61**
$RC_{2\text{-}04}$	3	798.12	Alexandre and Teodor, (2005)	3	879.82
$RC_{2\text{-}05}$	4	1300.25	Zbigniew and Piotr, (2001)	**5**	**1295.46**
$RC_{2\text{-}06}$	3	1152.03	Zbigniew and Piotr, (2001)	**4**	**1139.55**
$RC_{2\text{-}07}$	3	1061.14	Zbigniew and Piotr, (2001)	**4**	**1040.67**
$RC_{2\text{-}08}$	3	829.69	Rousseau *et al.*, (2002)	3	898.49

** Refer to the references for complete corresponding source entries*

Table 8 compares the routing performance between nine popular heuristics and HMOEA based on the average number of vehicles and average cost of routing in each category. In each grid, there are two numbers representing the average vehicle numbers (upper) and average cost of routing (lower), respectively. For example, in category C_1, the number pair (10.00, 838.00) means that over the 9 instances in C_1, the average vehicle numbers deployed is 10 and the average traveling distance is 838.00. The last row gives the total accumulated sum indicating the total number of vehicles and the total traveling distance for all the 56 instances. As can be seen, HMOEA leads to new best average results with the smallest CR and NV for category C_1. It also produces the smallest average routing cost for the categories of R_1, RC_1 and RC_2. The average number of vehicles for category R_1, is

2.7% higher as compared to the heuristics giving the second best average routing costs. Although the average routing cost of HMOEA is not the smallest for categories C_2 and R_2, the HMOEA only requires an average of 3.51 vehicles to serve all customers in the category of R_2, which is much smaller than the 5 vehicles that are required by the heuristic giving the best average routing cost in R_2. The results show that HMOEA performs equally well for both the objectives of *CR* and *NV*, which are optimized concurrently in the evolution. As shown in the last row of Table 8, HMOEA also provides the best total accumulated routing cost for the Solomon's 56 data sets.

Table 8 Performance comparison between different heuristics and HMOEA

Problem Class	Potvin and Bengio, (1996)	Taillard *et al.*, (1997)	Chiang and Russell, (1997)	Schulze and Fahle, (1999)	Bräysy and Gendreu, (2001b)	Ho *et al.*, (2001)	Tan *et al.*, 2001d	Tan *et al.*, 2001e	Lau *et al.*, 2003	HMOEA
C_1	10.00 838.00	10.00 828.45	10.00 828.38	10.00 828.94	10.00 828.38	10.00 833.32	10.00 851.96	10.00 841.96	10.00 832.13	10.00 **827.00**
C_2	3.00 589.90	3.00 590.30	3.00 591.42	3.00 589.93	3.00 589.86	3.00 593.00	3.20 620.12	3.00 611.2	3.00 589.86	3.00 590.00
R_1	12.60 1296.83	12.25 1216.70	12.17 1204.19	12.50 1268.42	11.92 1222.12	12.58 1203.32	13.20 1220.0	12.91 1205.0	12.16 1211.55	12.92 **1187.0**
R_2	3.00 1117.70	3.00 995.38	2.73 986.32	3.09 1055.90	2.73 975.12	3.18 951.17	4.40 985.69	5.00 929.6	3.00 1001.12	3.51 951.0
RC_1	12.10 1446.20	11.88 1367.51	11.88 1397.44	12.25 1396.07	11.5 1389.58	12.75 1382.06	13.30 1366.62	12.60 1392.3	12.25 1418.77	12.74 **1355.0**
RC_2	3.40 1360.60	3.38 1165.62	3.25 1229.54	3.38 1308.31	3.25 1128.38	3.75 1132.79	5.20 1108.50	5.80 1080.10	3.37 1170.93	4.25 **1067.00**
All	422 62572	416 57993	411 58502	423 60651	405 57710	432 57265	470 57903	471 56931	418 58476	441 **56262**

Fig. 24 shows the average simulation time (in seconds) for each category of data sets. The difference in computation time among the categories can be attributed to the flexibility of routing problem scenarios. From the statistics in Fig. 24, it is observed that all instances with longer time windows (i.e., category C_2, R_2 and RC_2) require a larger computation time. The reason is that these instances allow a more flexible arrangement in the routing plan since their time windows constraints are larger than other categories. Besides, a vehicle with longer route also takes up more computational time during the cost and feasibility evaluations process. Although HMOEA is capable of producing good routing solutions, it may require more computational time as compared with conventional approaches in order to perform the search in parallel as well as to obtain the globally optimized routing solutions (Tan *et al.*, 2002). Similar to most existing vehicle routing heuristics, the computational time should not be viewed as a major obstacle in solving the VRPTW, since HMOEA is developed for off-line simulation where the training time (computation time) is less important than the routing solutions. To reduce the computational time significantly, HMOEA is currently being integrated into the 'Paladin-DEC' distributed evolutionary computing framework (Tan *et al.*, 2002), where multiple inter-communicating subpopulations are implemented to share and distribute the routing workload among multiple computers over the Internet.

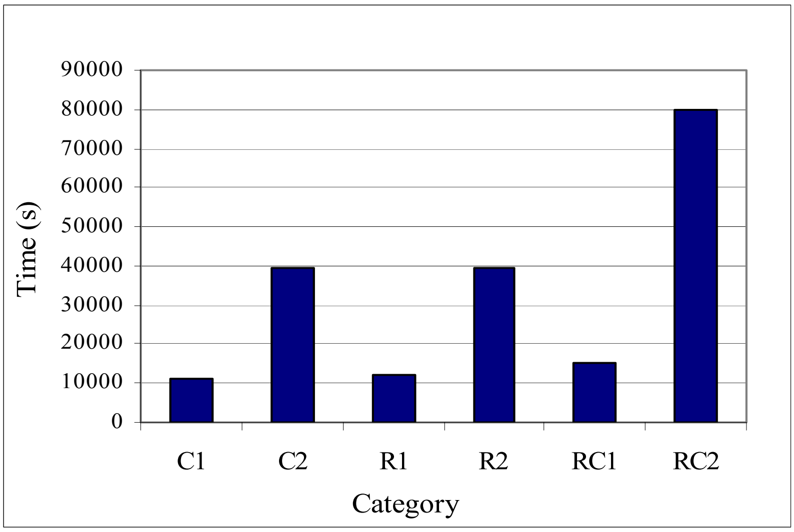
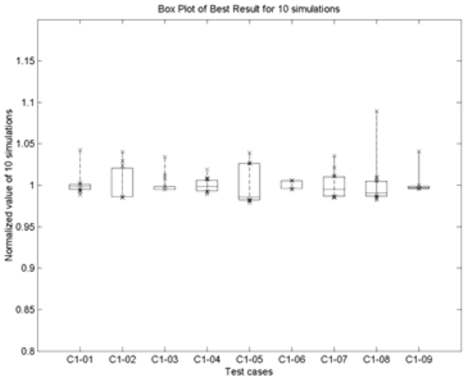


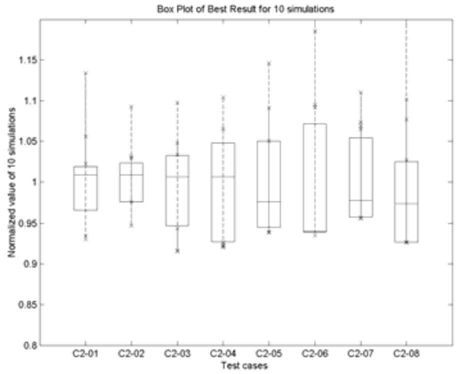
Figure 24 The average simulation time for each category of data sets

To study the consistency and reliability of the results obtained by HMOEA, 10 different but repeated simulations with randomly generated initial populations have been performed for the Solomon's 56 data sets. The simulation results are represented in box plot format (Chambers *et al.*, 1983) to visualize the distribution of simulation data efficiently. It should be noted that all the routing costs have been normalized to their mean values for easy comparisons among different test cases. Each box plot represents the distribution of a sample population where a thick horizontal line within the box encodes the median, while the upper and lower ends of the box are the upper and lower quartiles. The dashed appendages illustrate the spread and shape of distribution, while the dots represent the outside values. As shown in Fig. 25, the results obtained from HMOEA for the 10 different but repeated simulation runs are rather consistent and all variances are found to be within 5%-20% from the mean values. It is observed that the category of type 1 (C_1, R_1, RC_1) gives a smaller variance as compared to the category of type 2 (C_2, R_2,

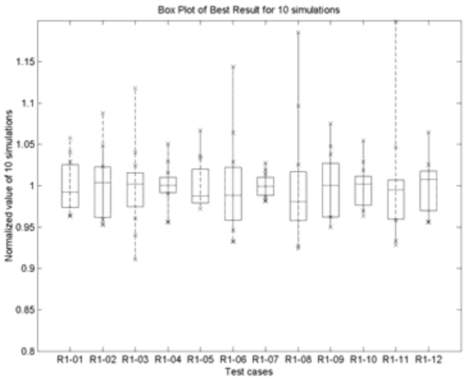
RC_2), since the number of customers per route (length of route) is shorter for the category of type 1, e.g., the possibility of variation in simulations is often larger for longer routes. Among all the categories, R_2 gives the largest variance, since the customers' locations are remotely located in this data set, i.e., a small difference in the routing sequence may result in significant changes to the solution.



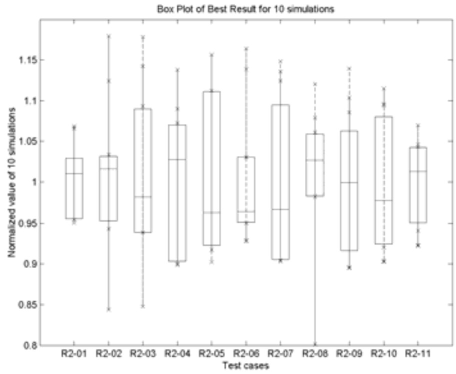
(a) C_1



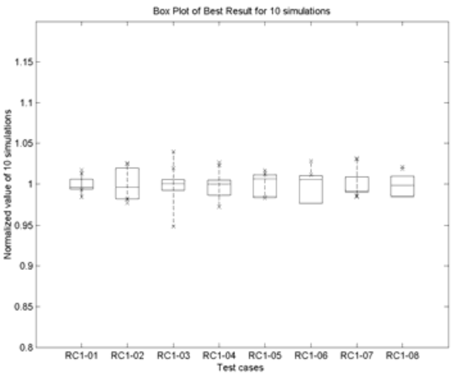
(b) C_2



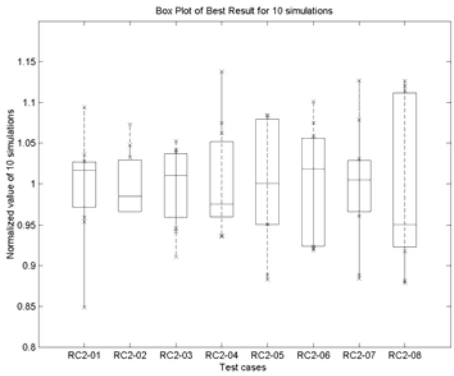
(c) R_1



(d) R_2



(e) RC_1



(f) RC_2

Figure 25 The variance in box plots for the Solomon's 56 data sets

In addition, Table 9 lists the means and standard deviations for the various simulation results as a supplement to the box plots above. From the table, similar observation can be found where results for category of type 1 (C_1, R_1, RC_1) have smaller standard deviation values as compared to others test cases. As all the test cases have various mean values, the last column was added to show the ratio (in percentage) between the standard deviation and the mean value so that difference between the test cases can be observed.

Table 9 Reliability performance for the algorithm

Test Case	Mean	Standard Deviation	Coefficient of Variation (%)
$C_{1\text{-}01}$	834.356	10.362	1.242
$C_{1\text{-}02}$	840.366	17.039	2.028
$C_{1\text{-}03}$	832.309	9.114	1.095
$C_{1\text{-}04}$	834.700	6.684	0.801
$C_{1\text{-}05}$	844.140	18.553	2.198
$C_{1\text{-}06}$	832.130	3.883	0.467
$C_{1\text{-}07}$	840.911	12.645	1.504
$C_{1\text{-}08}$	843.773	22.262	2.638
$C_{1\text{-}09}$	832.210	9.547	1.147
$C_{2\text{-}01}$	633.007	33.174	5.241
$C_{2\text{-}02}$	624.699	24.894	3.985
$C_{2\text{-}03}$	648.178	37.830	5.836
$C_{2\text{-}04}$	647.011	39.922	6.170
$C_{2\text{-}05}$	626.582	40.127	6.404
$C_{2\text{-}06}$	629.355	60.486	9.611
$C_{2\text{-}07}$	615.566	32.900	5.345

$C_{2\text{-}08}$	634.958	54.004	8.505
$R_{1\text{-}01}$	1674.750	52.578	3.139
$R_{1\text{-}02}$	1527.111	64.847	4.246
$R_{1\text{-}03}$	1239.951	61.951	4.996
$R_{1\text{-}04}$	1019.370	27.052	2.654
$R_{1\text{-}05}$	1414.421	42.145	2.980
$R_{1\text{-}06}$	1351.559	81.544	6.033
$R_{1\text{-}07}$	1100.593	16.203	1.472
$R_{1\text{-}08}$	1032.050	77.273	7.487
$R_{1\text{-}09}$	1218.848	49.281	4.043
$R_{1\text{-}10}$	1146.465	30.233	2.637
$R_{1\text{-}11}$	1139.025	80.838	7.097
$R_{1\text{-}12}$	1019.543	33.844	3.319
$R_{2\text{-}01}$	1268.992	56.082	4.419
$R_{2\text{-}02}$	1293.369	126.537	9.784
$R_{2\text{-}03}$	1102.993	119.548	10.838
$R_{2\text{-}04}$	878.510	78.007	8.880
$R_{2\text{-}05}$	1212.888	115.013	9.483
$R_{2\text{-}06}$	1013.004	79.507	7.849
$R_{2\text{-}07}$	942.896	93.956	9.965
$R_{2\text{-}08}$	986.284	99.927	10.132
$R_{2\text{-}09}$	1088.186	91.182	8.379
$R_{2\text{-}10}$	1087.685	85.863	7.894
$R_{2\text{-}11}$	879.473	46.375	5.273
$RC_{1\text{-}01}$	1667.535	16.778	1.006
$RC_{1\text{-}02}$	1496.692	28.038	1.873
$RC_{1\text{-}03}$	1336.273	30.617	2.291
$RC_{1\text{-}04}$	1177.408	19.424	1.650
$RC_{1\text{-}05}$	1590.388	18.74591	1.178
$RC_{1\text{-}06}$	1403.891	24.556	1.749
$RC_{1\text{-}07}$	1226.745	21.950	1.789

$RC_{1\text{-}08}$	1150.906	15.166	1.318
$RC_{2\text{-}01}$	1337.207	83.479	6.243
$RC_{2\text{-}02}$	1169.479	44.876	3.837
$RC_{2\text{-}03}$	1085.006	50.537	4.658
$RC_{2\text{-}04}$	916.533	60.125	6.560
$RC_{2\text{-}05}$	1362.118	107.403	7.885
$RC_{2\text{-}06}$	1236.963	85.496	6.912
$RC_{2\text{-}07}$	1153.294	82.895	7.188
$RC_{2\text{-}08}$	978.440	98.500	10.067

4.5 Conclusions

Vehicle routing problem with time windows (VRPTW) is inherently a multiobjective optimization problem that involves the optimization of routes for multiple vehicles in order to satisfy a set of constraints and to minimize multiple objectives, such as traveling distance and number of vehicles. A hybrid multiobjective evolutionary algorithm (HMOEA) has been proposed in this research, which incorporates various heuristics for local exploitation in the evolutionary search and the concept of Pareto's optimality for solving multiobjective optimization in VRPTW. The proposed HMOEA has been featured with specialized genetic operators and variable-length chromosome representation to accommodate the sequence-oriented optimization in VRPTW.

Unlike most conventional routing heuristics, this research is among the first to incorporate multiobjective optimization paradigm in solving the VRPTW. Without the need of aggregating multiple criteria and constraints of VRPTW into a compromise function, the HMOEA optimizes all routing constraints and objectives concurrently, which improves the routing solutions in many aspects, such as lower routing cost, wider scattering area, and better convergence trace. Extensive simulations have been performed on the benchmark Solomon's 56 VRPTW 100-customer instances, which yielded 20 routing solutions better than or equivalent to the best solutions published in literature.

Chapter 5 Truck and Trailer Vehicle Scheduling Problem

This research considers a transportation problem for moving empty or laden containers for a logistic company. Owing to the limited resource of its vehicles (trucks and trailers), the company often needs to subcontract certain job orders to outsourced companies. A model for this truck and trailer vehicle scheduling problem (TTVSP) is first constructed in the research. The solution to the TTVSP consists of finding a complete routing schedule for serving the jobs with minimum routing distance and number of trucks, subject to a number of constraints such as time windows and availability of trailers. To solve such a multiobjective and multi-modal combinatorial optimization problem, a hybrid multiobjective evolutionary algorithm (HMOEA) featured with specialized genetic operators, variable-length representation and local search heuristic is applied to find the Pareto optimal scheduling solutions for the TTVSP. Detailed analysis is performed to extract useful decision-making information from the multiobjective optimization results as well as to examine the correlations among different variables, such as the number of trucks and trailers, the trailer exchange points, and the utilization of trucks in the routing solutions. It has been shown that the HMOEA is effective in solving multiobjective combinatorial optimization problems, such as finding useful trade-off solutions for the TTVSP routing problem.

5.1 The Trucks and Trailers Vehicle Scheduling Problem

Singapore ranks among the top international maritime centers of the world. Its sheltered and deep-water harbor lies strategically at the crossroads of major sea routes in South-east Asia. It is the focal point for some 400 shipping lines with links to more than 740 ports worldwide. The Republic's standing as an international maritime centre rests on its port, which is one of the busiest in the world in terms of container throughput. In 2002, the port handled a total of 16.94 million twenty-foot equivalent units (TEUs) (Maritime, 2002). In order to support the port activities in lieu with the extremely high throughput at the port, container related logistic services are very prosperous in Singapore. A general model for vehicle capacity planning system (VCPS) consisting of a number of job orders to be served by trucks and trailers daily was constructed for a logistic company that provides transportation services for container movements within the country (Lee *et al*., 2003). Due to the limited capacity of vehicles owned by the company, engineers in the company have to decide whether to assign the job orders of container movements to its internal fleet of vehicles or to outsource the jobs to other companies daily. The Tabu search meta-heuristic was applied to find a solution for the VCPS problem, where some new rules on how to assign jobs for outsourcing were derived and shown to be about 8% better than existing rules adopted by the company (Lee *et al*., 2003).

By analyzing different kinds of job orders received from the company, this research presents a transportation solution for trucks and trailers vehicle scheduling problem (TTVSP) containing multiple objectives and constraints, which is extended from the VCPS model with detail maneuver of trailers in a routing plan. In TTVSP, the trailers are resources with certain limitations similar to real world scenarios and the allocation of trailers in different locations could affect the routing plans. The TTVSP is a difficult problem which involves many intricate factors such as time window constraints and availability of trailers. The number of trucks in a fleet regulates the maximum number of jobs that can be handled internally within a certain period of time and all jobs must be serviced within a given time window. Instead of handling jobs by the internal fleet of trucks, the jobs can also be considered for outsourcing, if necessary. The routing plans in TTVSP also needs to determine the number of trailer exchange points (TEPs) that are distributed in the region where different type of trailers can be found. Besides, there are a wide variety of job orders that may have diverse requirements for the types of the trailers, time window constraints as well as locations of the source and destination.

The transportation solution to TTVSP contains useful decision-making information, such as the best fleet size to accommodate a variety of job orders and the trend for different number of trailers available at TEPs, which could be utilized by the management to visualize the complex correlations among different variables in the routing problem. Dynamic resource management is an essential component in a logistic company. Long term planning in resource management (such as the number of vehicles) is rather tedious especially when the business is in a dynamic environment. In order to maintain efficiency, minimizing the cost and investment and maximizing quality of service, long term resource planning and day-to-day operations are two crucial factors to ensure an organization’s success. In this research, various test cases for the TTVSP model are generated with random variables simulating the long-term operation of business activities. The management can thus formulate the planning for certain variables, such as the number of trucks (long term capital cost) so that the day-to-day operational cost could be kept at the minimum.

5.1.1 Variants of Vehicle Routing Problems

Vehicle routing problem (VRP) is a generic name referred to a class of combinatorial problem in which customers are to be served by a number of vehicles. Some famous models in literature for vehicle routing problems include Gendreau *et al*., (1999a), Laporte *et al*., (2002), Belenguer *et al*., (2000), Yang *et al*., (2000), Kenyon and Morton, (2003), Ichoua *et al*., (2003), Ghiani and Improta (2000), Swihart and Papastavrou (1999), Salhi and Sari (1997), Min *et al*., (1998) and Wu *et al*., (2002). Among these models, there are three types of vehicle routing problems closely related to the TTVSP model presented in this research, i.e., vehicle routing problem with time windows (VRPTW), vehicle scheduling problem (VSP), and truck and trailer routing problem (TTRP).

The vehicle routing problem with time windows (VRPTW) diverts from the famous vehicle routing problem (VRP). In this problem, a set of vehicles with limited capacity is to be routed from a central depot to a set of geographically dispersed customers with known demands and predefined time window. The time window can be specified in terms of single-sided or double-sided window. In single-sided time window, the pickup points usually specify the deadlines by which they must be serviced. In double-sided time window, however, both the earliest and the latest service times are imposed by the nodes. A vehicle arriving earlier than the earliest service time of a node will incur waiting time. This penalizes the transport management in either the direct waiting cost or the increased number of vehicles, since a vehicle can only service fewer nodes if the waiting time is longer. Some recent publications of VRPTW can be found in Bräysy (2003), Breedam (2001), Caseau and Laburthe (1999), Dullaert (2000), Gezdur and Türkay (2002), Ioannou *et al*. (2001), Shaw (1998), Li and Lim (2002), Chavalitwongse *et al*. (2003), Bent and Van (2001) and Berger *et al.* (2001). Surveys about VRPTW can be found in Desrosier *et al*., (1995), Desrochers *et al*., (1992), Golden and Assad (1988), Solomon (1987), Kilby *et al*., (2000), Toth and Vigo (2002), Bräysy and Gendreau (2001a, 2001b) etc. In contrast to the TTVSP, the VRPTW neither have any limitation on resources of trailers nor the outsourcing of jobs to external companies.

The vehicle scheduling problem (VSP) (Baita *et al*., 2000; Brandão and Mercer, 1997; Pretolani, 2000; Boland *et al*., 2000; Dror, 2000; Hertz and Mittaz, 2001) assumed that the routing to different sites can be completed with multiple trips. Each trip consists of a pair of specified source and destination, each one defined by the starting and ending times. The objective is to minimize the number of vehicles and the cost function based upon deadheading trips (gas, driver etc) and idling time for the vehicle. The constraints for this model include the traveling distance and time for normal service and refueling as well as the restriction that certain tasks can only be handled by specified type of vehicles. In contrast to vehicle routing problem, one customer may be visited more than once or not at all, which is solely depending on the trips data. Although trips in VSP may be analogous to the concept of a job in TTVSP, the VSP does not include the complexity of trailer type constraints.

Chao (2002) presented the problem of TTRP (a variant of VRP), which considers the fleet size of trucks and trailers in the model. In order to provide service to different categories of customers, there are three types of routes in a solution: (1) route that a truck travels alone (2) route that a truck and trailer are required (3) route that trailer is only required at certain sub-tour. The objective is to minimize the total traveling distance and the cost incurred by the fleet. Unlike TTRP, the TTVSP requires the trucks to visit trailer exchange points for picking up the correct trailer types depending on the jobs to be serviced. Besides, jobs that are not routed by self-fleets in TTVSP can be outsourced to external companies.

5.1.2 Meta-heuristic Solutions to Vehicle Routing Problems

Most vehicle routing problems are NP-hard and associated with real world transportation problems (Glaab, 2002; Baptista *et al.*, 2002; Mourão and Almeida, 2000; Dillmann *et al.*, 1996; Fölsz *et al.*, 1995; Karkazis and Boffey, 1995; Muyldermans *et al.*, 2002; Doerner *et al.*, 2002; Baita *et al.*, 2000). Due to the inherent variations in real world environment, the solution to each vehicle routing problem is often unique and satisfies an exclusive set of constraints and objectives according to the problem scenario. Generally, vehicle routing problems have been attempted by different approaches ranging from exact algorithms (Applegate *et al.*, 2002; Bard *et al.*, 2002; Mingozzi *et al.*, 1999) to heuristics (Gerdeseen, 1996; Kohl *et al.*, 1999; Beullens *et al.*, 2003; Renaud and Boctor, 2002; Breedam, 2002; Toth and Vigo, 1999; Liu and Shen, 1999; Beasley and Christofides, 1997). Categorized by Fisher (1995) as the third generation approach, a number of meta-heuristics such as Tabu search (Taillard *et al*., 1997; Kelly and Xu, 1999; Rego, 1998; Gendreau *et al*., 1999c; Tuzun and Burke, 1999; Amberg *et al*., 2000; Rego and Roucairol, 1995; Potvin *et al*., 1996; Cordeau *et al*., 2001; Cordone and Wolfler, 2001; Lee *et al*., 2003), ant colony optimization (Gambardella *et al*., 1999; Reimann and Doerner, 2002), simulated annealing (Breedam, 1995; Chiang and Russel, 1996) and genetic algorithms (Gehring and Homberger, 2001; Grefenstette *et al*., 1985; Homberger and Gehring, 1999; Malmborg, 1996; Poon and Carter, 1995; Tan *et al*., 2001a; 2001b; Thangiah *et al*., 1994; Thangniah, 1995) have been applied to find good solutions for large-scale vehicle routing problems. A recent survey on various meta-heuristic algorithms was presented by Ribeiro and Hansen (2002).

The TTVSP problem addressed in this research is NP-hard, which involves the optimization of routes for multiple trucks in order to meet all given constraints and to minimize multiple objectives of routing distance and number of trucks concurrently. Some of the existing routing approaches that strive to minimize a single criterion of routing cost or number of trucks is not suitable for solving such a multi-modal and mutltiobjective combinatorial problem. The TTVSP should be best tackled by multiobjective optimization methods, which offer a family of Pareto-optimal scheduling solutions containing both the minimized routing cost and number of trucks. In this research, a hybrid multiobjective evolutionary algorithm (HMOEA) that incorporates the heuristic search for local exploitation and the concept of Pareto's optimality for finding the trade-off is applied to solve the problem of

TTVSP. The HMOEA optimizes all routing constraints and objectives concurrently, without the need of aggregating multiple criteria into a compromise function. Unlike conventional multiobjective evolutionary algorithms (MOEAs) that are designed with simple coding or genetic operators for parameterized optimization problems (Cvetkovic and Parmee, 2002; Knowles and Corne, 2000; Tan *et al*., 2001c), the HMOEA is featured with specialized genetic operators and variable-length chromosome representation to accommodate the sequence-oriented optimization problem in TTVSP.

The research is organized as follows: Section 5.2 describes the scenario and modeling of the TTVSP with mathematical formulation. Section 5.3 gives a brief description of multiobjectve evolutionary optimization and its applications in a number of domain-specific combinatorial problems. The program flowchart of HMOEA and its various features including variable-length chromosome representation, specialized genetic operators, Pareto fitness ranking and local search heuristics are also described in Section 5.3. Section 5.4 presents the extensive simulation results and discussions for the TTVSP problem. Conclusions are drawn in Section 5.5.

5.2 The Problem scenario

The TTVSP model with detail maneuver of the trailers in a routing plan is extended from a real world VCPS system proposed by Lee *et al.,* (2003). Both of the problems are variants of vehicle routing problem with time windows constraints

(VRPTW). The additional constraints and conditions apply in TTVSP indicate that the problem is fundamentally more difficult than a simple VRPTW, and thus it is essentially another NP hard problem. In solving the TTVSP, the movement of containers among customers, depots and the port are major transportation job orders considered. A container load is handled like a normal truckload but these loads use containers with a possible chassis instead of trailers only. From the equipment assignment point of view, a correct trailer type is essential for the routing. For an inbound job, a loaded container is taken from a vessel to a customer and returned empty to the depot. For an outbound job, however, an empty container is picked up from the depot and taken to the customer before returning loaded to the vessel. Every job order contains the location of source and destination as well as other customers' information. Other specification such as load requirement and time windows are specified as hard constraints in the model. There are a total of 6 types of job orders which are varied according to the source and destination (port, warehouse, depot or trailer exchange), time windows (tight or loose), loaded trip (or empty) and type of trailers (20 or 40) as follows:

- Import with trailer type 20
- Import with trailer type 40
- Export with trailer type 20
- Export with trailer type 40
- Empty container movement with trailer type 20
- Empty container movement with trailer type 40

The logistic company owns a maximum of 40 trucks and a number of trailers that are larger than the number of trucks. A truck must be accompanied with a trailer when servicing a customer, i.e., the routing needs to consider both the locations of truck and trailer. An "export" job order works as follows: a truck first picks up a correct trailer at a trailer exchange point and a container at the depot. It then proceeds to the warehouse and leaves the trailer and container there for about 2 days where the container is filled. A truck (which may not be the same truck as earlier) will later be allocated to move the loaded container using the earlier assigned trailer and leaves the container at the port before departing with the trailer. In contrast, an "import" job order works as follows: a truck picks up a correct trailer at a TEP before it proceeds to the port. The trailer is used to carry loaded container at the port. The truck then moves the container to the warehouse and leaves it there for about 2 days. A truck (which may not be the same truck as earlier) will later move this empty container from the warehouse to the depot (using a trailer) and leaves the depot with its trailer unloaded. Intuitively, there are times when a truck has a correct trailer type and thus can serve a job without going to a trailer exchange point. Otherwise, a truck is required to pick up a trailer (from the nearest TEP where the trailer is available to be picked up or exchanged) when it has mismatch trailer type or does not carry a trailer. The number of trailers available at an exchange point depends on how many trailers were picked up and returned to the TEP. The constraint imposed on the model is the time windows at the source and destination of job orders. An assumption is made such that all trailer exchange points have similar operating hours as the truck drivers' working hours, i.e., from 8:00 am to 8:00 pm.

5.2.1 Modeling the Problem Scenarios

Based on the scenarios described, some refinements have been made to the model proposed by Lee *et al.,* (2003). The problem is modeled here on a daily basis where the planning horizon spans only one day. All import and export jobs consist of two sub-trips and a two-day interval at the customer warehouses. Therefore the two-day interval at customer warehouses divides a job nicely into two separate planning horizons (one day each). The import and export jobs can be broken into two independent tasks, where each of them falls into a different planning horizon. In this way, job orders are broken into sub-job type precisely (Hereinafter this is referred as sub-job or a task). Generally a task involves traveling from a point (source) to another point (destination) as listed in Table 10.

Table 10 The task type and its description

Task type	Task description	Source	Destination	Trailer type
1	Sub-trip of import job	Port	Warehouse	20
2	Sub-trip of import job	Port	Warehouse	40
3	Sub-trip of import job	Warehouse	Depot	20
4	Sub-trip of import job	Warehouse	Depot	40
5	Sub-trip of export job	Depot	Warehouse	20
6	Sub-trip of export job	Depot	Warehouse	40
7	Sub-trip of export job	Warehouse	Port	20
8	Sub-trip of export job	Warehouse	Port	40
9	Empty container movement	Port	Depot	20
10	Empty container movement	Depot	Port/Depot	20
11	Empty container movement	Port	Depot	40
12	Empty container movement	Depot	Port/Depot	40

The number of trailers at TEPs depends on the trailers that are left over from the previous planning horizon. All the pickup, return and exchange activities can also change the number of trailers available. Besides, a number of trailers could also be parked at the customer warehouses instead of the TEPs. All these undetermined factors suggest that the resource of trailers available at each TEP at the initial of planning horizon is random. Therefore the daily number of trailers at each trailer exchange point is randomly generated in our model. A truck has to pick up a correct trailer from the nearest TEP if it serves task type 1, 2, 5, 6, 9, 10, 11 or 12 and does not have a trailer or has an incorrect trailer type. For task type 3, 4, 7 or 8, the truck does not need to visit a TEP before servicing the task since the correct trailer has been brought to the place in advanced. In contrast, trucks that serve sub-job type 3, 4, 7 or 8 must not have any trailers. In this case, if a trailer is attached to the truck, it must be returned to a trailer exchange point before servicing the task. For example, a truck that serves sub-job type 7 leaves the destination (port) of a previous task with a trailer. If the same truck is to serve another task type 3, 4, 7 or 8, it must travel to a TEP to drop the trailer obtained previously. In brief, a truck is required to visit a trailer exchange point under the following conditions:

- It needs a trailer for task type 1, 2, 5, 6, 9, 10, 11 or 12 and it does not have a trailer.
- It needs a trailer for task type 1, 2, 5, 6, 9, 10, 11 or 12 and it has an incorrect trailer type.
- It has a trailer but it has to service sub-job type 3, 4, 7 or 8, e.g., the truck needs to travel to a TEP for dropping the trailer before servicing the task.

Obviously the availability of trailers at TEPs should be updated frequently since the number of trailers changes with the pick-up and return activities, e.g., a trailer that is returned earlier in a day will be available for pick-up later in the same day. To model these activities, the approach of time segmentation for trailer resources is used as follows:

- Working hours per day: 12 hours × 60 mins = 720 mins
- Time per segment: 10 mins
- Number of time slots available: $720/10$ slots = 72 slots

Hence the number of trailers available for pick-up in a particular time slot is equal to the number of trailers in previous time slot, added by the trailers returned in previous time slot and deducted the trailers picked up in previous time slot. In this approach, different trailer types are managed and updated in separate lists. For example, a TEP has 3 trailers (with type 20) and the following events occur in the current time slot: one trailer (type 20) is returned and two trailers (type 20) are picked up. In this case, the trailer exchange point should have two trailers (type 20) available for pick up in the next time slot.

5.2.2 Mathematical Model

Decision Variables:

$X_{ik_m} \in \{0,1\}$, where $i = \{1,...,I\}$, $k = \{1,...,K\}$, $m = \{1,...,M\}$. If task i is assigned to truck k as the m^{th} task, $X_{ik_m} = 1$, otherwise $X_{ik_m} = 0$;

$X_{i0} \in \{0,1\}$, $i \in \{1,..,I\}$. If task i is subcontracted to companies, $X_{i0} = 1$, otherwise $X_{i0} = 0$.

Parameters:

I = Number of tasks;

K = Maximum number of trucks;

M = Maximum number of jobs that can be handled by one truck in a planning horizon;

J = Number of trailer exchange points;

y = Task type, i.e., $y \in \{1,...,12\}$;

$I(y)$ = The set of task with type y;

$\bigcup_y I(y) =$ All tasks = $\{1,...I\}$;

TW = time segment for trailer resources = 10;

MTW = maximum number of time slots = 72.

<u>Symbol</u>

$\lceil x \rceil$: The smallest integer larger or equal to x;

$\lfloor x \rfloor$: The largest integer smaller or equal to x.

Distance of tasks' location

D_{hji}: Distance from destination of previous task h to trailer point j followed by source of task i;

D_{hi}: Distance from destination of previous task h to source of task i;

D_i: Distance from source of task i to destination of task i.

Task handling time

H_{i1}: Handling time at source of task i;

H_{i2}: Handling time at destination of task i.

Task time window

R_{i0} : Start-time at the source of task *i*;

R_{i1} : End-time at the source of task *i*;

R_{i2} : Start-time at the destination of task *i*;

R_{i3} : End-time at the destination of task *i*;

A_{k0} : Start available time for truck *k*;

A_{kf} : End available time for truck *k*.

Cost

P_i : Routing cost of task *i* for internal fleet operation;

S_i : Routing cost of task *i* for outsourced;

<u>Number of trailers at trailer exchange point</u>

TP_{40j} : Initial number of trailer type 40 at point *j*;

TP_{20j} : Initial number of trailer type 20 at point *j*.

Minimization Objectives:

The scheduling solutions should minimize both the criteria of routing cost and the number of trucks concurrently as follows:

$$\text{Routing cost} = \sum_{i=1}^{I}\sum_{k=1}^{K}\sum_{m=1}^{M} X_{ikm} P_i + \sum_{i=1}^{I} X_{i0} S_i \ ;$$

$$\text{Number of trucks} = \sum_{k=1}^{K} \left\lceil \frac{\sum_{i=1}^{I}\sum_{m=1}^{M} X_{ik_m}}{I} \right\rceil .$$

subject to the following requirements and constraints:

Task and trailer types requirements

$$pickup20_{k,m} = \sum_{y=1,2,4,5,6,8,11,12} \sum_{y'=1,5,9,10} \sum_{j \in I(y)} \sum_{i \in I(y')} (X_{ik_m})(X_{jk_{m-1}}) \qquad for\ m > 1;$$

$$pickup20_{k,m} = \sum_{y=1,5,9,10} \sum_{i \in I(y)} X_{ik_m} \qquad for\ m = 1;$$

$$pickup40_{k,m} = \sum_{y=1,2,3,5,6,7,9,10} \sum_{y'=2,6,11,12} \sum_{j \in I(y)} \sum_{i \in I(y')} (X_{ik_m})(X_{jk_{m-1}}) \qquad for\ m > 1;$$

$$pickup20_{k,m} = \sum_{y=2,6,11,12} \sum_{i \in I(y)} X_{ik_m} \qquad for\ m = 1;$$

$$return20_{k,m} = \sum_{y=3,7,9,10} \sum_{y'=2,3,4,6,7,8,11,12} \sum_{j \in I(y)} \sum_{i \in I(y')} (X_{ik_m})(X_{jk_{m-1}}) \qquad for\ m > 1;$$

$$return40_{k,m} = \sum_{y=4,8,11,12} \sum_{y'=1,3,4,5,7,8,9,10} \sum_{j \in I(y)} \sum_{i \in I(y')} (X_{ik_m})(X_{jk_{m-1}}) \qquad for\ m > 1;$$

$visit_{k,m} \in \{0,1\}$;

$pick20_{k,m} + pick40_{k,m} + return20_{k,m} + return20_{k,m} \leq 2visit_{k,m}$;

Single assignment

A task is only assigned to one truck k (as the m^{th} task) or outsourced to other companies,

$$\sum_{k=1}^{K} \sum_{m=1}^{M} X_{ik_m} + X_{i0} = 1 \text{ for } i \in \{1,\ldots,I\}$$

Jobs must be assigned sequentially

For $k \in \{1,...,K\}$, $m \in \{1,...,M-1\}$, $\sum_{i=1}^{I} X_{ik_{(m+1)}} \leq \sum_{i=1}^{I} X_{ik_{(m)}}$

Time sequence for each task

For $k \in \{1,...,K\}$, $m \in \{1,...,M-1\}$, $T_{k_{(m+1)}(0)} = T_{k_{(m)}(2)}$;

For $k \in \{1,...K\}$, $m \in \{1,...M\}$,

$$T_{k_m(1)} \geq T_{k_m(1)} + \sum_{i=1}^{I} X_{ik_m} \{H_{i1} + \sum_{h=1}^{I} X_{hk_{m-1}} [visit_{k,m} D_{hji} + (1 - visit_{k,m}) D_{hi}]\};$$

$$T_{k_m(2)} \geq T_{k_m(1)} + \sum_{i=1}^{I} X_{ik_m} (D_i + H_{i2}).$$

Time window constraints

For $k \in \{1,...,K\}$, $m \in \{1,...,M-1\}$, $A_{k0} \leq T_{k_m(0)} \leq A_{kf} - (T_{k_m(2)} - T_{k_m(0)})$;

For every particular $i \in \{1,...,I\}$, $\begin{array}{l} R_{i0} \leq \sum_{k=1}^{K}\sum_{m=1}^{M} X_{ik_m}(T_{k_m} - H_{i1}) + X_{i0}R_{i0} \leq R_{i1} \\ R_{i2} \leq \sum_{k=1}^{K}\sum_{m=1}^{M} X_{ik_m}(T_{k_m(2)} - H_{i2}) + X_{i0}R_{i2} \leq R_{i3} \end{array}$.

Trailer constraints

$X_{ik_m}(t) \in \{0,1\}$, where $X_{ik_m}(t) = 1$ when the event falls into time window *t*,

$$X_{ik_m}(t) = 1 - \left\lceil \frac{\left| \left(\left\lfloor \frac{T_{k_m(1)}}{TW} \right\rfloor - t \right) \right|}{MTW} \right\rceil$$

The number of trailer type 20 at time slot *t* = 0, i.e., $B_{20j}(0) = TP_{20j}$;

For every *t* = 0 to 71, and every *j*, the number of trailer type 20 available for next time slot, *t* +1, is,

$$B_{20j}(t+1) = B_{20j}(t) + \sum_{i=1}^{I}\sum_{k=1}^{K}\sum_{m=1}^{M} X_{ik_m}(t) return20_{k,m} -$$

$$\sum_{i=1}^{I}\sum_{k=1}^{K}\sum_{m=1}^{M} X_{ik_m}(t) pickup20_{k,m},$$

where $B_{20j}(t) \geq 0$.

The number of trailer type 40 at time slot *t* = 0, i.e., $B_{40j}(0) = TP_{40j}$;

For every t = 0 to 71, and every j, the number of trailer type 40 available for next time slot, t +1, is,

$$B_{40j}(t+1) = B_{40j}(t) + \sum_{i=1}^{I}\sum_{k=1}^{K}\sum_{m=1}^{M} X_{ik_m}(t) return40_{k,m} -$$

$$\sum_{i=1}^{I}\sum_{k=1}^{K}\sum_{m=1}^{M} X_{ik_m}(t) pickup40_{k,m} ,$$

where $B_{40j}(t) \geq 0$.

5.2.3 Test Cases Generation

The TTVSP models various factors affecting the routing performance, particularly on the importance of trailer resources such as the trailers allocation in multiple trailer exchange sites and the location of trailer exchange points. In order to examine these factors thoroughly, a number of test cases with different combination of variables are generated according to the following criteria:

- Number of tasks
- Total number of trailers
- Number of trailers and allocation
- Number of trailer exchange points (with trailer resources assigned initially)

The test cases are generated based on the scenario of one-day activity for a logistic company. The jobs schedule starts from 8:00 am to 8:00 pm (12 hours a day). All the tasks must be finished within a day and the details of every task are generated. The service map for the problem contains one port, three depots and five trailer exchange points. The five TEPs are named as TEP1, TEP2. TEP3, TEP4 and TEP5, which are located at disperse places and may have different initial number of trailers. The problem also defines the location of 80 customer sites spreading across the area randomly. The service map for the problem is a 120×120 grid and the locations of customers are given as a pair of (x, y) coordinates. The distance (traveling time) among any two points is calculated as 0.5×(triangular distance), where the value of 0.5 is merely a scaling factor such that a truck can serve around 3 tasks per day in average. The timing constraint is also specified in the test cases, e.g., the handling time at the source and destination (i.e., port, depot, and customer warehouses) requires 10 minutes, which must be included in calculating the time needed for a complete job handling. The time windows for the source and destination of each job are generated according to the type of jobs. The availability of trailer resources is quantified into 10-minute slots. The return of a trailer is only visible to others after the current time slot, where the retrieval of a trailer gives immediate effect to the current count of trailers. The cost for each task type is based on the way tasks are accomplished, i.e., by self-fleet service or outsourced to external companies. There is no hard rule to specify whether the cost for internal fleet is cheaper than outsource fleet and vice versa, i.e., the cost merely depends on the type of jobs to be served.

There are a total of 28 test cases generated in this study, which differs in terms of the number of task orders, the number of trailers, allocation of trailers, and the number of trailer exchange points. However, information about customer warehouses and other important locations like port and depots remains unchanged.

Table 11 lists the test cases for NORM (Normal) category, where the trailers are allocated "equally" to TEPs. As shown in Table 11, the test cases in this category are divided into 4 groups with different number of tasks in the range of 100 to 132, and all TEPs can contribute to the supply of any demands for trailers. As shown in Table 12, the 8 test cases for TEPC (Trailer Exchange Point Case) category contain a constant of 132 tasks, but are assigned with extreme trailer allocation strategies. In some cases, only one TEP is allocated with trailers, while the available number of trailers remains constant at 30 for all test cases in this category. As shown in Table 13, the LTTC (Less Trailer Test Case) category comprises of 8 test cases with an equal number of trailers. In this category, the available number of trailers is set as 10, e.g., the trailer resources for both TEPC and LTTC test cases share the same distribution ratio but are assigned with different quantity of trailers.

Table 11 Test cases for the category of NORM

Group	Test case*	Job number	Trailers at Each TEP	Teps allocated with trailers	Distribution
100	test_100_1_2	100	1 or 2	5	uniform
	test_100_2_3	100	2 or 3	5	uniform
	test_100_3_4	100	3 or 4	5	uniform
112	test_112_1_2	112	1 or 2	5	uniform
	test_112_2_3	112	2 or 3	5	uniform
	test_112_3_4	112	3 or 4	5	uniform
120	test_120_1_2	120	1 or 2	5	uniform
	test_120_2_3	120	2 or 3	5	uniform
	test_120_3_4	120	3 or 4	5	uniform

132	test_132_1_2	132	1 or 2	5	uniform
	test_132_2_3	132	2 or 3	5	uniform
	test_132_3_4	132	3 or 4	5	uniform

*The last digit denotes the number of trailers allocated for each TEP

Table 12 Test cases for the category of TEPC

Test case	**Job number**	**Number of trailers at TEPs**	**TEPs allocated with trailers**	**Distribution***
test_132_tep5	132	30	5	uniform
test_132_tep1a	132	30	1	TEP1
test_132_tep1b	132	30	1	TEP2
test_132_tep1c	132	30	1	TEP3
test_132_tep1d	132	30	1	TEP4
test_132_tep1e	132	30	1	TEP5
test_132_tep3a	132	30	3	Distributed among TEP1, TEP3 and TEP5
test_132_tep3b	132	30	3	Distributed among TEP1, TEP2 and TEP4

*Fixed number of trailers and different distribution of TEPs

Table 13 Test cases for the category of LTTC

Test case	**Job number**	**Number of trailers at TEPs**	**TEPs allocated with trailers**	**Distribution***
test_132_ltt5	132	10	5	uniform
test_132_ltt1a	132	10	1	TEP1
test_132_ltt1b	132	10	1	TEP2
test_132_ltt1c	132	10	1	TEP3

test_132_ltt1d	132	10	1	TEP4
test_132_ltt1e	132	10	1	TEP5
test_132_ltt3a	132	10	3	Distributed among TEP1, TEP3 and TEP5
test_132_ltt3b	132	10	3	Distributed among TEP1, TEP2 and TEP4

*Less trailers and different distribution of TEPs

5.3 A Hybrid Multiobjective Evolutionary Algorithm

As described in the Introduction, the TTVSP should be best solved via multiobjective optimization, e.g., it involves optimizing routes for multiple trucks to meet all constraints and to minimize the conflicting costs of routing distance and number of trucks concurrently. The HMOEA applied in for solving TTVSP problem is similar to the HMOEA in chapter 4 with some minor modification to adapt the TTVSP problem. In general the main program flow is similar to the proposed HMOEA. Both the problem can use the same initialization flow. The explanation below highlights some of the difference of the proposed algorithm in solving this particular problem.

5.3.1 Variable-Length Chromosome Representation

The chromosome in an evolutionary algorithm is often represented as a fixed-structure bit string and the bits position in a chromosome are usually assumed to be independent and context insensitive. However, such a representation is not suitable for the order-oriented combinatorial TTVSP problem, for which the sequence among customers is essential. In HMOEA, a variable-length chromosome representation is adopted, where each chromosome encodes a complete routing plan including the number of routes and tasks served by the trucks, e.g., a route is a sequence of tasks to be served by a truck. In every route there must be at least one task assignment, and any task that is not assigned to a route is considered for outsourcing (all the outsourced tasks are contained in a list). The number of trailers must be up-to-date and a routing plan must include supplementary information of trailers availability in every trailer exchange points. As shown in Fig. 26, a chromosome may consist of several routes and each route or gene is not a constant but a sequence of tasks to be served. Such a variable-length representation is efficient and allows the number of trucks to be manipulated and minimized directly for the multiobjective optimization in TTVSP.

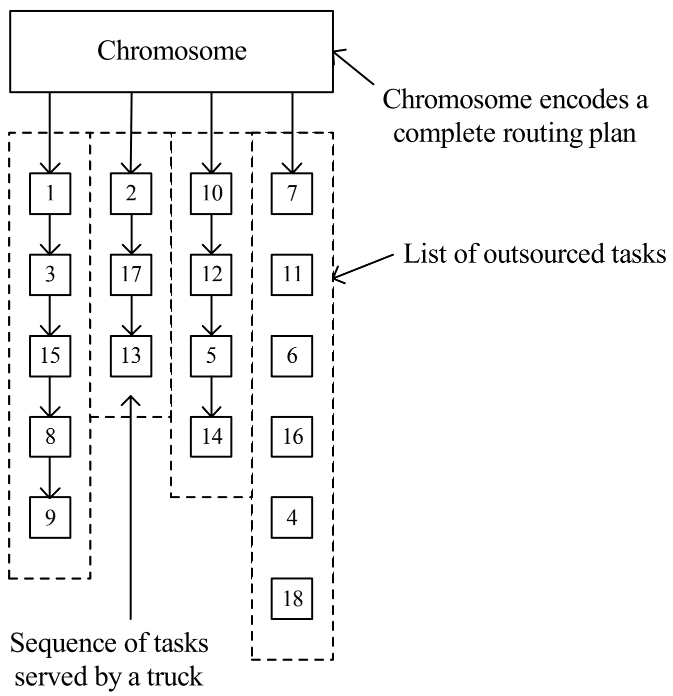


Figure 26 The data structure of chromosome representation in HMOEA

5.3.2 Multimode Mutation

Gendreau *et al*., (1999b) proposed a RAR mutation operator, which extracts a node and inserts it at a random point of the routing sequence in order to retain the feasibility of solutions. Ishibashi *et al*. (2000) extends the approach to a shift mutation operator, which extracts a segment or a number of nodes (instead of a node) and inserts it at a new random point to generate the offspring. During the crossover operation by HMOEA, routes' sequence is exchanged in a whole chunk and no direct manipulation is made to the internal ordering of the nodes for TTVSP. The sequence in a route is modified only when any redundant nodes in the chromosome are deleted. A multimode mutation is adopted in HMOEA, which serves to complement the crossover by optimizing the local route information of a chromosome. The mutation is expected to trigger changes of tasks sequence within a chromosome and the mutation rate is considerably small since it could be destructive to the chromosome structure and information of routes. A random number is generated to choose between two possible operations in the mutation. The first operation picks two routes in a chromosome randomly and concatenates the first route to the second route before deleting the first route from the chromosome. In the second operation, the sequence containing all the outsourced tasks is evaluated as a new route. The approach also checks feasibility on the route in order to delete any tasks that cause violation to any of the constraints, and those deleted tasks will be considered as outsourced tasks.

5.3.3 Fitness Sharing

A simple fitness sharing (Fonseca and Fleming, 1998) is incorporated in HMOEA to prevent genetic drift, which is a phenomenon where a finite population tends to settle on a single optimum even if many other local optima exist. The fitness sharing models the competitions among individual for finite resource available in a niche. When the number of individuals in its neighborhood increases, the fitness of an individual is degraded as a result of the competition. The sharing approach measures the niching distance in the objective domain to achieve diversity of solutions on the tradeoff curve. The niche radius, σ, is a parameter that defines the size of neighborhood where all individuals within this distance would contribute towards the sharing function. The distance between individuals is normalized to the maximum range of objective space, which is dynamically computed at each generation. Let $dist(x,y)$ be the normalized distance between individual x and individual y, the sharing function sh can be defined as follows,

$$sh\left(dist(x,y)\right) = \begin{cases} \left(1 - dist(x,y)/\sigma\right)^2 & \text{if } dist(x,y) < \sigma \\ 0 & \text{otherwise} \end{cases} \qquad (1)$$

The sharing value of an individual will be increased by other individuals that are found located within the niche radius and the sharing value is higher when the distance between the individuals is shorter. With the help of sharing function, the niche count nc is defined as,

$$nc(x) = \sum_{y \in individuals} sh\left(dist(x,y)\right) \qquad (2)$$

During the tournament selection, individuals with a lower rank in partial order will be selected for reproduction, where the partial order ranking between two individuals depends on both their Pareto rank and niche counts. Rigorously, the partial order $\geq_p$ for two individuals i and j is defined as,

$$i \geq_p j \text{, if } \left[rank(i) > rank(j)\right] \text{ or } \left[rank(i) = rank(j) \text{ and } nc(i) > nc(j)\right]$$

5.4 Computational Results

The HMOEA was programmed in C++ based on a Pentium III 933 MHz processor with 256 MB RAM under the Microsoft Windows 2000 operating system. From the empirical results of preliminary experiments, we found that HMOEA performed equally well with small changes of parameter values. As the general rules of thumb, the crossover rate is relatively larger than mutation rate. The choice is reasonable as high mutation rate tends to destroy the good chromosomes and preventing the preservation of good parents. Table 14 shows the parameter settings chosen after some preliminary experiments. These settings should not be regarded as an optimal set of parameter values, but rather a generalized one for which the HMOEA performs fairly well over the test problems.

Table 14 Parameter settings for the simulations

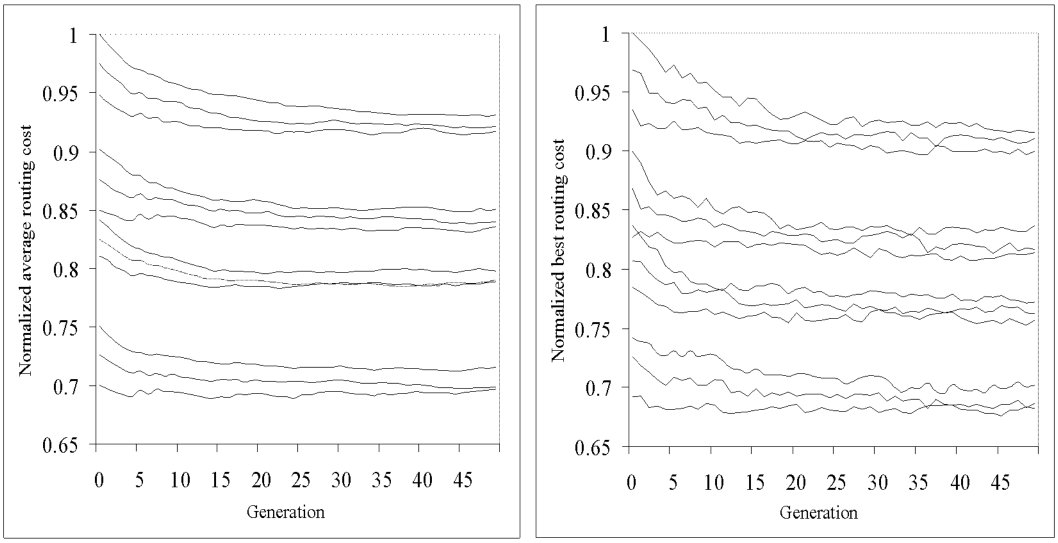
PARAMETER	VALUE
Crossover rate	0.8
Mutation rate	0.3
Population size	800
Generation size	1000 or no improvement over the last 5 generations
Niche radius	0.04

This section contains the computational results and analysis of optimization performances for all problem instances. Section 5.4.1 studies the performance of convergence trace and Pareto-optimality for multiobjective optimization using the 12 test cases in normal category. In the same section, several other performance metrics such as the utilization rate, the progress ratio and a simple scenario of using the results of the routing plan are included. Section 5.4.2 analyzes the optimization problem when different trailer allocation scenarios happen based on the test cases of TEPC and LTTC (each of the TEPC and LTTC categories contains 8 test cases). In Section 5.4.3, the optimization performance of HMOEA is compared with two other multiobjective evolutionary algorithms based upon various performance measures.

5.4.1 Multiobjective Optimization Performance

5.4.1.1 Convergence Trace

Convergence trace is an important performance indicator to show the effectiveness of an optimization algorithm. The two objectives in TTVSP are the number of trucks and the routing cost as defined in Section 5.2. Fig. 27 shows the normalized average and best routing costs at each generation for the 12 test cases in normal category, where each line represents the convergence trace for each of the test cases. As can be seen, the routing costs decline nicely as the evolution proceeds. The same observation can be found in Fig. 28, where the normalized average number of trucks at each generation is plotted. The rapid reduction of the number of trucks in Fig. 28 is expected as the initial population in HMOEA was generated randomly.



(a) (b)

Figure 27 Convergence trace of the normalized average (a) and best (b) routing costs

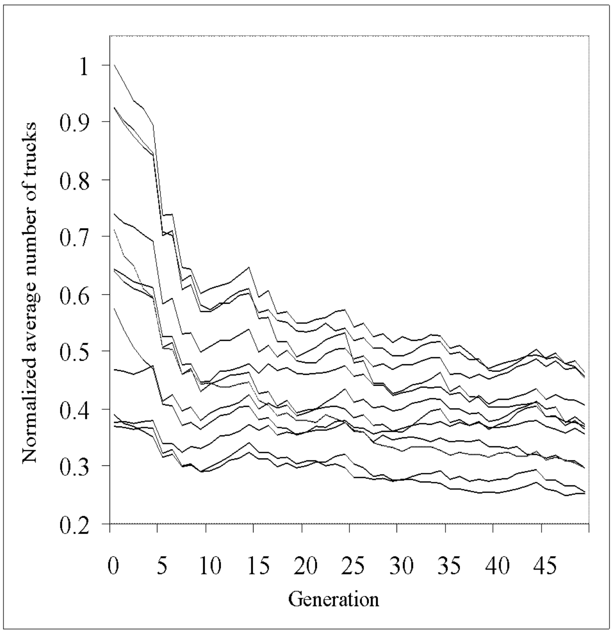


Figure 28 Convergence trace of the normalized average number of trucks

5.4.1.2 Pareto Front

In solving a vehicle scheduling problem, the logistic manager is often interested in not only getting the minimum routing cost, but also the smallest number of trucks required to service the plan. In order to reduce the routing cost, more number of trucks is often required and vice versa, i.e., the two criteria are noncommensurable and often competing with each other. Fig. 30 shows the evolution progress of Pareto front for all the 12 test cases in normal category. Fig. 29 is a zoom-in version of the one of the test case; all others enlarged figures are attached in Appendix 2. In the simulation, the largest available vehicle number is limited to 35, which is more than sufficient to cater the number of tasks in each test case. The various Pareto fronts obtained at the initial generation (First), two intermediate generations (Int 1 and Int 2) and the final generation (Final) are plotted in Fig. 30 with different markers. As can be seen, there is only a small number of non-dominated solutions appeared at the initial generations, which are also congested at a small portion of the solution space. However, as the evolution proceeds, the diversity of the population increases significantly and the non-dominated solutions gradually evolve towards the final trade-off curve. A dashed line connecting all the final non-dominated solutions is drawn for each test case in Fig. 30, which clearly shows the final trade-off or routing plan obtained by the HMOEA. It should be noted that the Pareto front includes the plan with zero truck number that subcontracts all tasks to external company, although such a policy is apparently not practical to adopt because it is against the will of the logistic management.

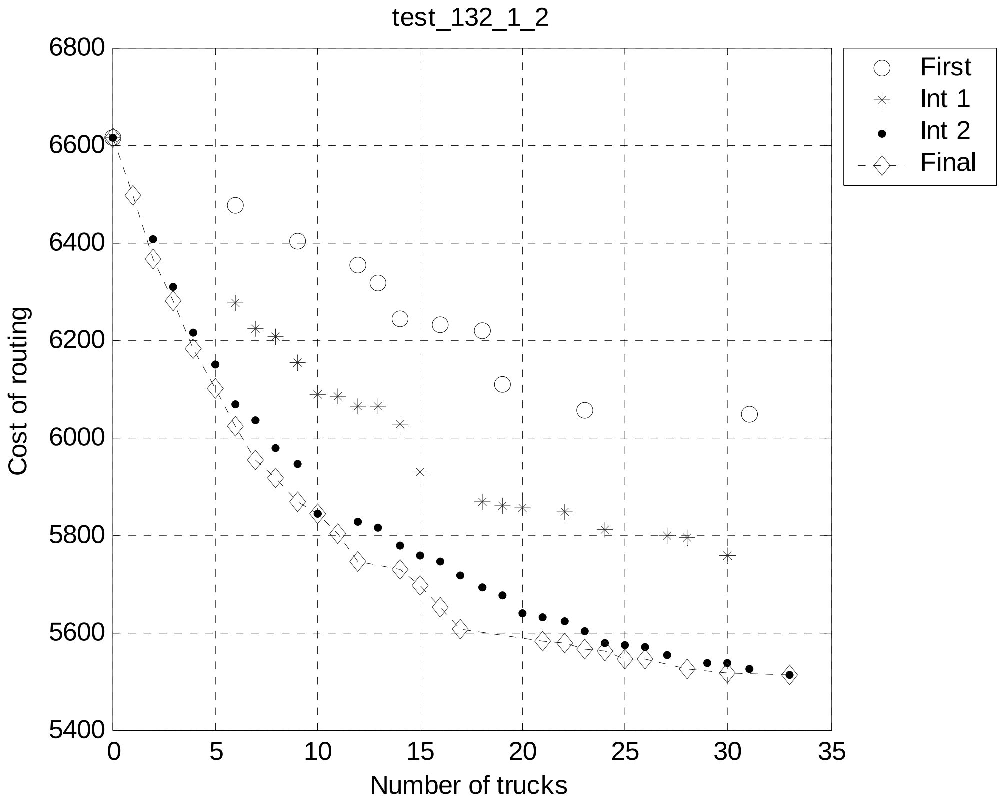


Figure 29 Zoom in for evolution progress of Pareto front

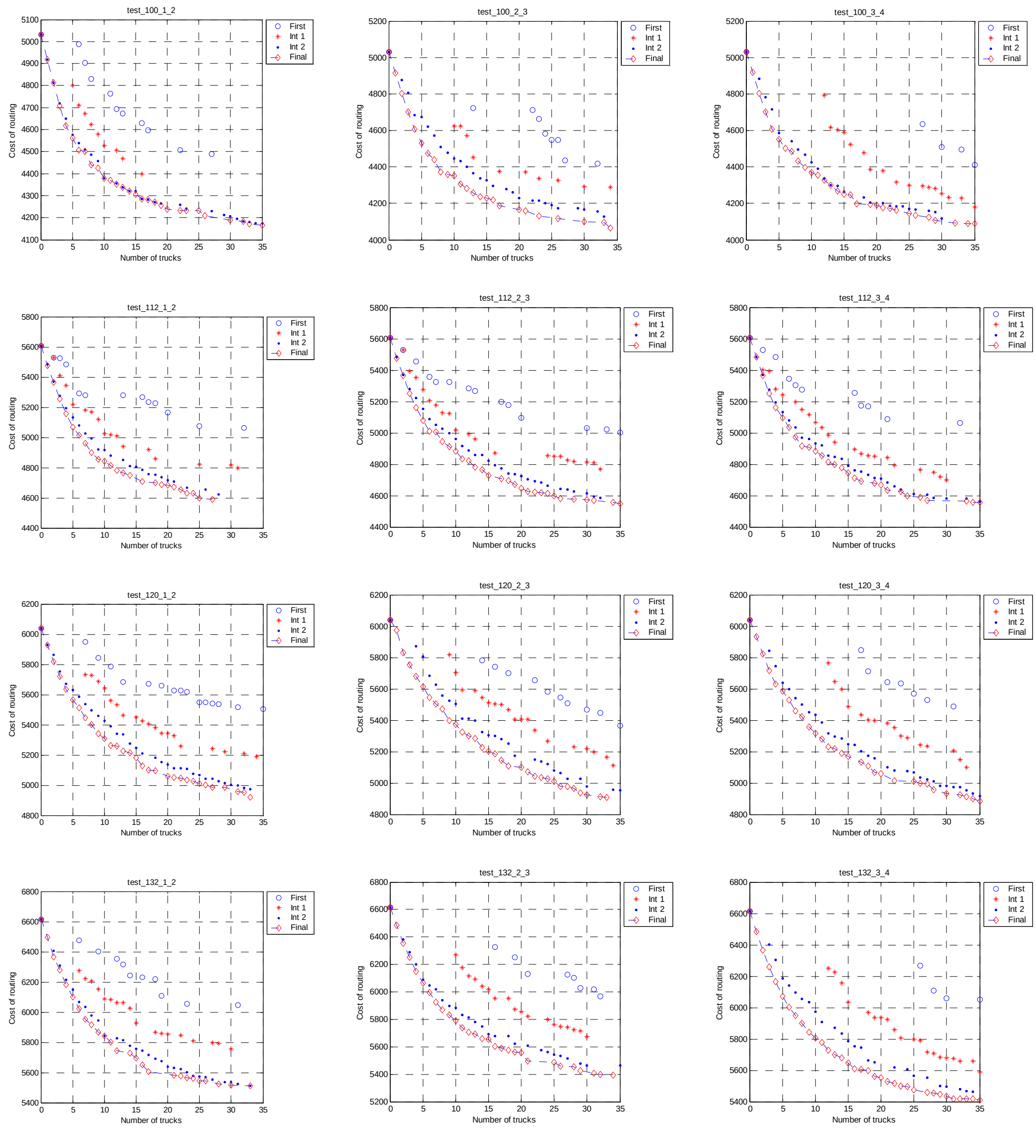


Figure 30 The evolution progress of Pareto front for the 12 test cases in normal category

5.4.1.3 Routing Plan

The average best routing cost for each truck number of the 12 test cases in normal category is plotted in Fig. 31, which shows an obvious trade-off between the two objectives of routing cost and truck number in TTVSP. This trade-off curve is useful for the decision-maker to derive an appropriate routing schedule according to the current situation. The information about the number of tasks to be serviced and the number of trailers available at each trailer exchange point is often available. Based on the information, if the number of trucks available in a company is fixed, the logistic manager can estimate the required routing cost from the trade-off curve in Fig. 31. In contrast, if the manager is given a specified budget or routing cost, he or she can then determine the minimum number of internal trucks to be allocated so that the spending can be kept below the budget. For example, if the routing cost is to be kept below 5100, then the company must allocate at least 10 trucks for serving the task orders. However, if only 15 trucks are allocated by the company, then the incurred routing cost would be around 4900 to 5000, including the cost payment for outsourced companies.

Fig. 32 shows the average progress ratio at each generation for the 12 test cases in normal category, which is a useful convergence measures for the Pareto front in multiobjective optimization. The progress ratio at any generation is defined as the domination of one population to another (Tan *et al.*, 2001c),

$$pr^{(n)} = \frac{nondom\_indiv^{(n)} \text{ dominating } nondom\_indiv^{(n-1)}}{nondom\_indiv^{(n)}} \quad (3)$$

As shown in Fig. 32, the average *pr* starts from a value close to one indicating the high probability of improvement to the solutions at the initial stage. As the evolution continues, the *pr* decreases to a small value which means that the evolution is nearly converged since the possibility of finding new improved non-dominating solution is low.

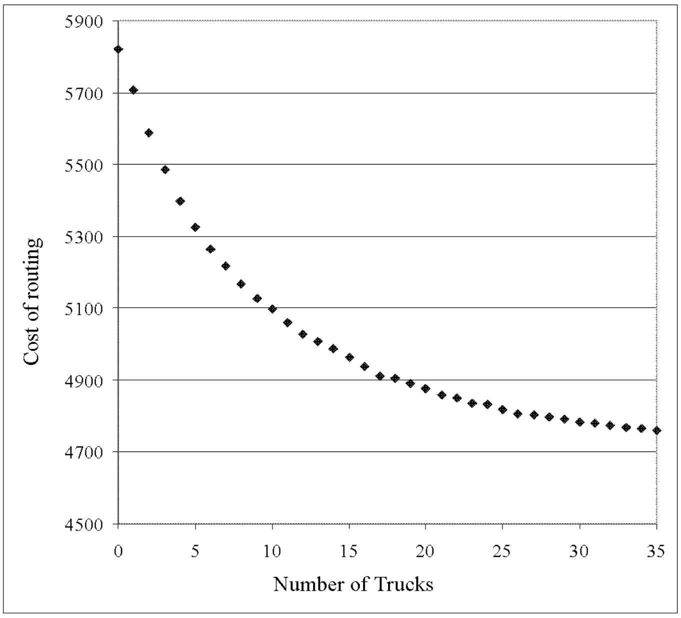


Figure 31 The trade-off graph between cost of routing and number of trucks

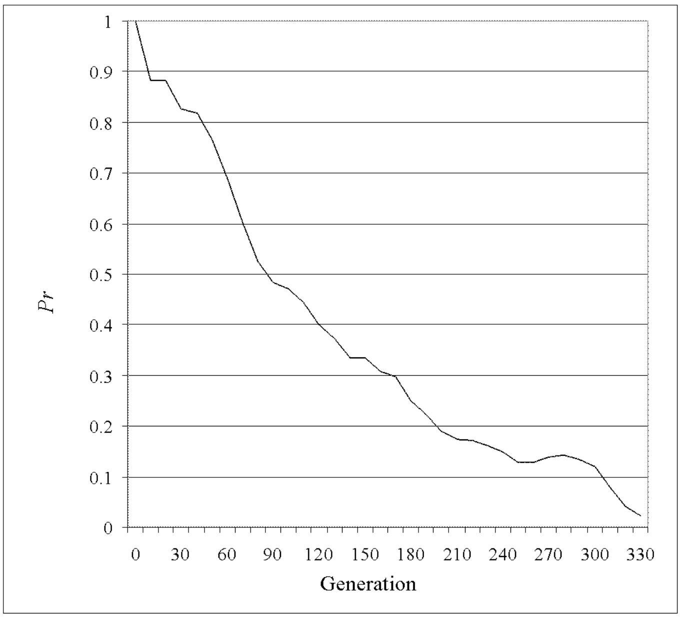


Figure 32 The average *pr* at each generation for the 12 test cases in normal category

5.4.1.4 Utilization of Trucks

Besides the trade-off curve, one interesting aspect to be investigated in TTVSP is the utilization of trucks, which is defined as the average number of tasks completed by a truck. A higher utilization means fewer trucks with smaller associated fixed cost are needed to perform the tasks. Fig. 33 shows the average utilization of all test cases in the normal category based on the non-dominated solutions at the initial (Initial) and final (Final) generation, respectively. Clearly, the utilization performance of TTVSP after the optimization by HMOEA has been improved consistently for different number of trucks.

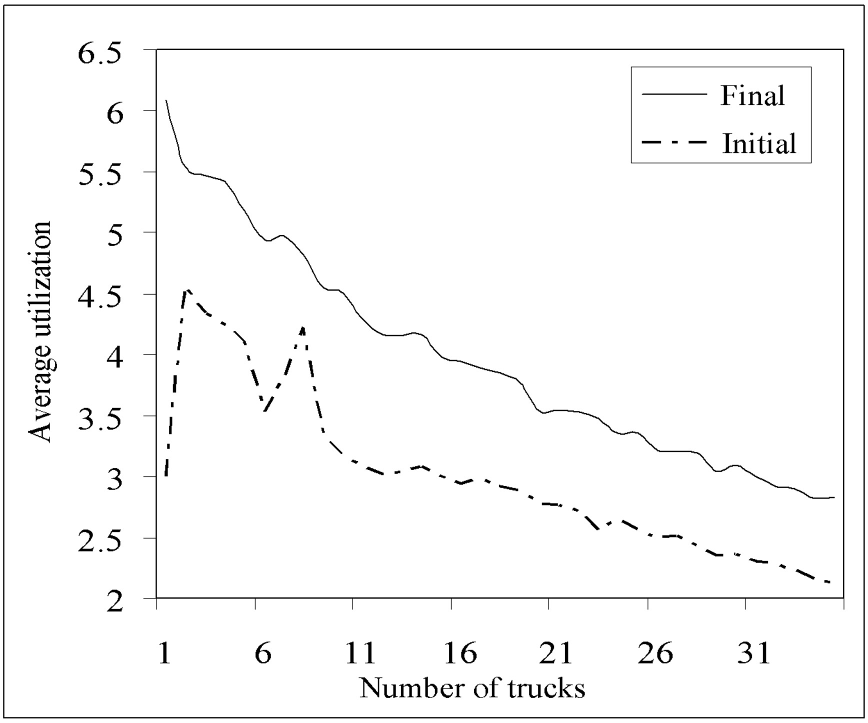


Figure 33 The average utilization of all test cases in the normal category

Fig. 34 shows that the utilization of trucks for all the test cases in normal category increases as the number of trailers increases, which is plotted by taking the average utilization of every individuals in the final population served by the internal fleets.

Besides the number of trucks employed, utilization performance in TTVSP is also correlated to the trailer allocation, e.g., the utilization is higher as the number of trailers increases, since abundant trailer resources help to reduce unnecessary traveling to farther TEPs as well as to eliminate infeasibility caused by the trailer constraints in TTVSP. Obviously, such information is useful for the management to achieve a high utilization performance in TTVSP before arriving at the final routing plan.

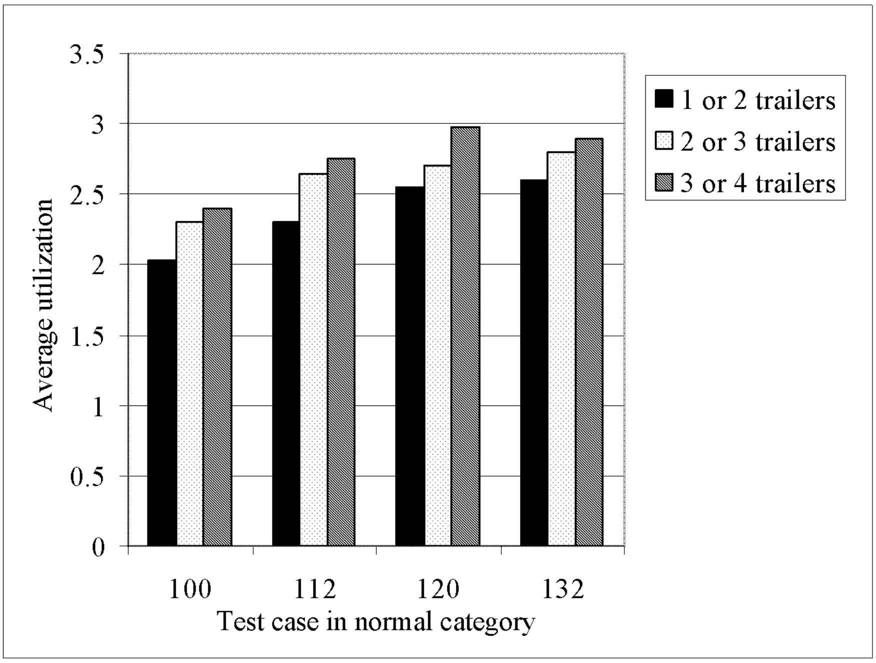


Figure 34 The average utilization of all individuals in the final population

5.4.2 Computational Results for TEPC and LTTC

The test cases in TEPC and LTTC categories are designed to examine situations such as excessive or inadequate trailer resources. The following sub-sections study the extreme trailers allocation policy: Section 5.4.2.1 compares the optimization results of TEPC and LTTC while Section 5.4.2.2 investigates the effects of the number and location of trailer exchange points in TTVSP.

5.4.2.1 Scenario of Extreme Trailer Allocation

The resource of trailers is one of the key elements in TTVSP. For the normal test category, since the variation of trailer number at TEPs is small and the tasks that require trailers are only a proportion of the total tasks, the effect of trailer number to routing cost is insignificant as discussed in Section 5.4.1. In this sub-section, the scenario of excessive and limited trailer resources is compared based on the test cases in TEPC (with 30 trailers) and LTTC (with 10 trailers) categories. Fig. 35 shows the box plot of routing costs for the final non-dominated solutions in different test cases of TEPC and LTTC categories. Each box plot represents the distribution of a sample set where a vertical line within the box encodes the median, while the right and left ends of the box are the upper and lower quartiles. Dashed appendages illustrate the spread and shape of distribution, and dots represent the outside values. In the figure, 132_tep1 and 132_ltt1 represents the combined result for the test cases with only one TEP for TEPC and LTTC, respectively. As can be seen, the mean routing costs for test cases in TEPC are consistently lower than the cases in LTTC. When the number of trailers is abundant as in TEPC, a feasible solution can be found more easily as compared to LTTC where resource of trailers is limited and the search for better solutions is restricted by the lack of trailers. The results show that the trailers and their distribution greatly affect the final scheduling performance. It is thus important to have enough trailers allocation at the initial of planning horizon, and a good routing policy should favor the choice that brings more trailers back to TEPs at the end of each planning horizon.

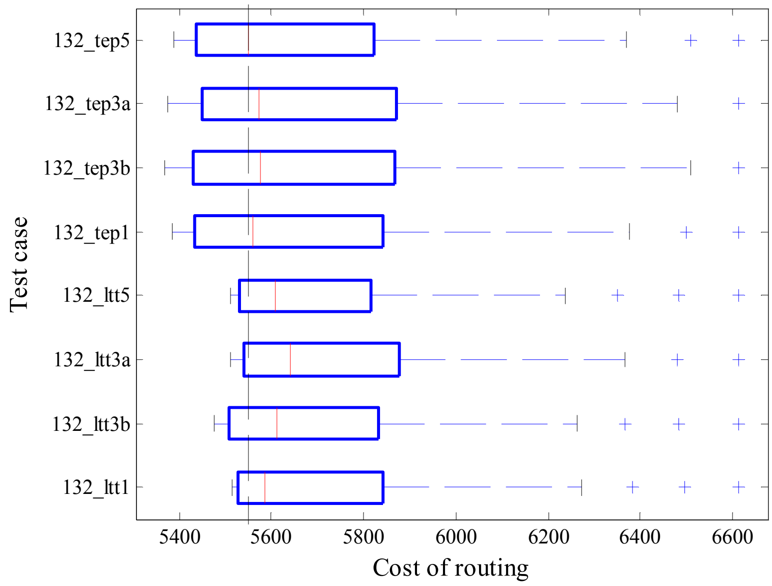


Figure 35 The performance comparison of abundant TEPC with limited trailers in LTTC

5.4.2.2 The Number and Location of TEPs

This sub-section compares the routing performance among the different test cases within each category of TEPC and LTTC. Fig. 36 and Fig. 37 show the box plots of routing costs for the final non-dominated solutions in different test cases of TEPC and LTTC, respectively. In Fig. 36, the mean value of test_132_tep5 is extended vertically and chosen as a reference since this test case has its trailer resources distributed uniformly to all the TEPs. It can be seen that the range of routing costs for the various test cases is rather closed to test_132_tep5. In addition, there is only minor difference in terms of the mean routing cost, except for the case of test_132_tep1e where the trailers are allocated to only one TEP. Hence the location of TEP is not strategic for TTVSP. Similarly, the mean routing cost of test_132_ltt_1e is also inferior as compared to other test cases in the LTTC category as shown in Fig. 37. The results suggest that the final destinations of trailers should be properly planned and allocated at suitable TEPs that support the routing for the next planning horizon.

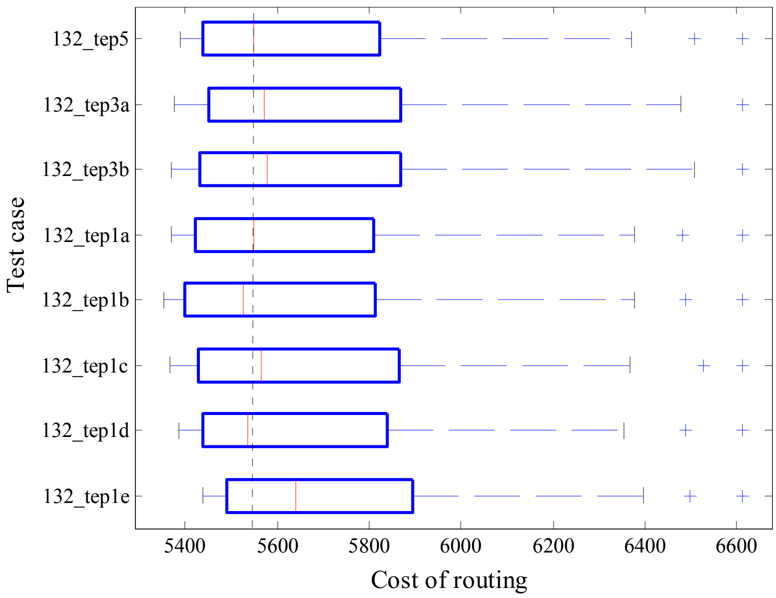


Figure 36 The performance comparison of different test cases in TEPC category

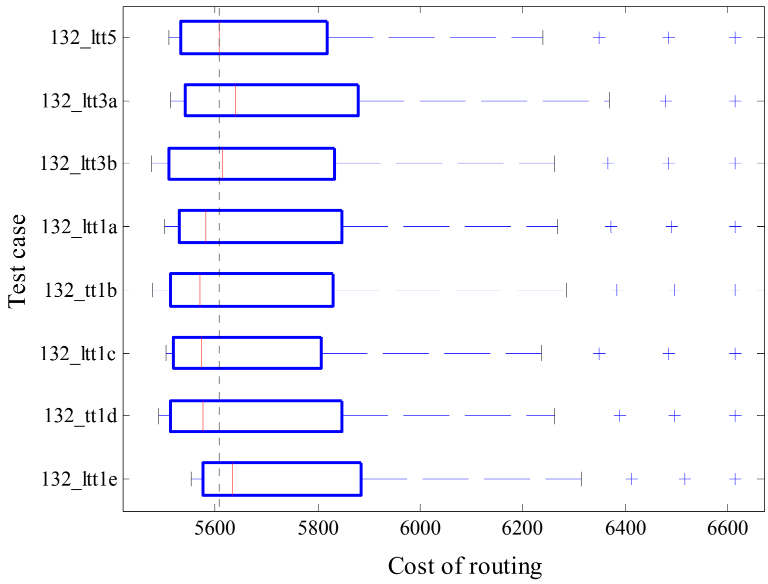


Figure 37 The performance comparison of different test cases in LTTC category

5.4.3 Comparison Results

In this section, the performance of HMOEA is compared with two variants of evolutionary algorithms, i.e., MOEA with standard genetic operators as well as MOEA without hybridization of local search. The comparison allows the effectiveness of the various features in HMOEA, such as specialized genetic operators and local search heuristics, to be examined. The multiobjective evolutionary algorithm with standard generic operators (STD_MOEA) includes the commonly known cycle crossover and RAR mutation. The cycle crossover is a general crossover operator that preserves the order of sequence in the parent partially and was applied to solve the traveling salesman problems by Oliver *et al.* (1987). The remove and reinsert (RAR) mutation operator removes a task from the sequence and reinsert it at a random position (Gendreau *et al.*, 1999b). The multiobjective evolutionary algorithm without hybridization of local search

(NH_MOEA) employs the specialized genetic operators in HMOEA but excludes the local search heuristic. The experimental setups and parameter settings of STD_MOEA and NH_MOEA are similar to the settings of HMOEA in Table 14.

5.4.3.1 Average Routing Cost

To compare the quality of solutions produced by the algorithms, the average routing cost (ARC) of the non-dominated solutions in the final population is calculated for various test cases with different number of tasks as shown in Fig. 38. In the figure, the average value of ARC is plotted for each group of the test cases with equal number of tasks in the normal category. As can be seen, the STD_MOEA that employs standard genetic operators incurs the highest ARC since its operators are not tailored made for the TTVSP problem. According to the no free lunch theorem (Wolpert and Macready, 1996), any optimization methods should be tailored to the problem domain for best performance. The results in Fig. 38 also illustrate that the HMOEA outperforms NH_MOEA and STD_MOEA consistently, which produces the lowest routing cost for all test cases. The average routing cost of the non-dominated solutions in the final population for test cases in the category of TEPC and LTTC is shown in Fig. 39 and Fig. 40, respectively, where a similar outstanding optimization performance for HMOEA is observed.

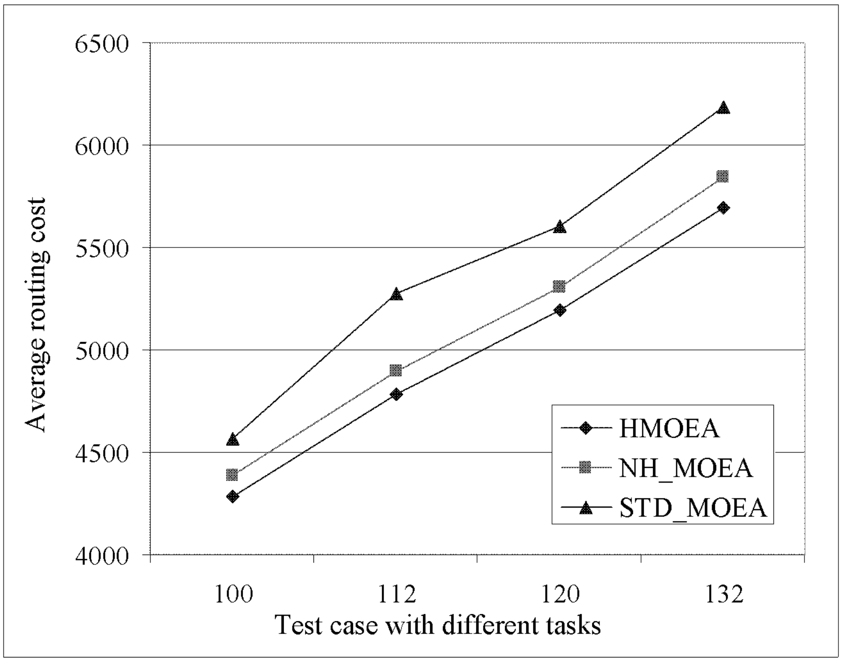


Figure 38 The average routing cost for the normal category

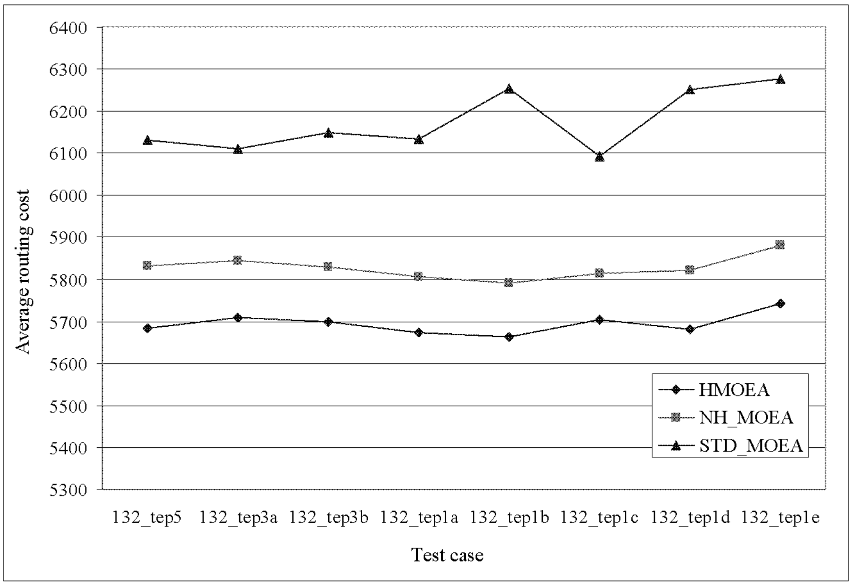


Figure 39 The average routing cost for the TEPC category

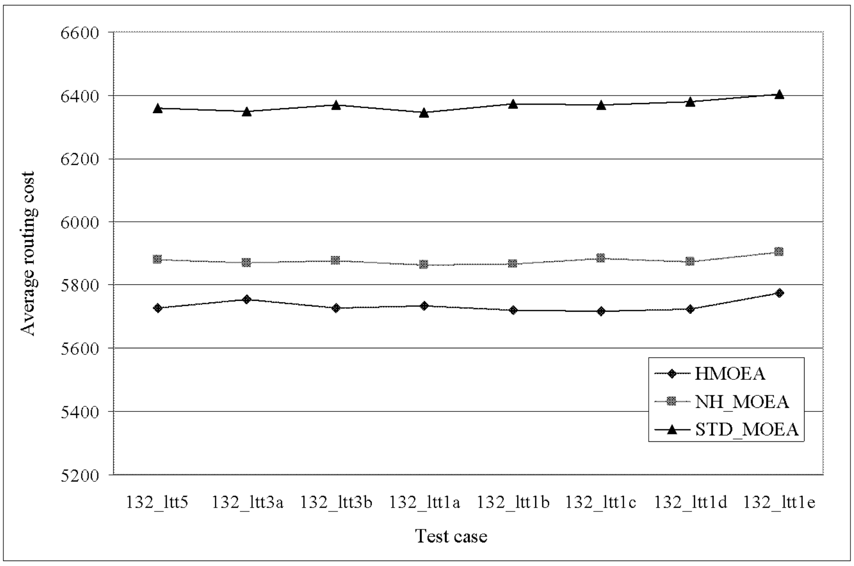


Figure 40 The average routing cost for the LTTC category

5.4.3.2 Ratio of Non-dominated Individuals

In multiobjective optimization, it is often desired to find as many as possible useful candidate solutions that are non-dominated in a given population, which could be measured by the ratio of non-dominated individuals (RNI) as proposed by Tan *et al.*, (2001c). For a given population *X*, the RNI in percentage is formulated as,

$$RNI(X)\% = \frac{nondom\_indiv}{N} \times 100\% \quad (4)$$

where *nondom_indiv* is the number of non-dominated individuals in population *X*, and *N* is the size of the population *X*. Without loss of generality, Fig. 41 shows the RNI for the three algorithms based on a randomly selected test case 132_3_4. As can be seen, the RNI value of STD_MOEA is the lowest among the three algorithms. The evolution in STD-MOEA stopped at around 90 generations as no improvement was observed for 5 generations continuously. The results also show that the search performance of HMOEA for non-dominated solutions is slightly better than

NH_MOEA. Besides, the HMOEA also has the best average RNI of 1.89 as compared to the value of 1.71 and 0.44 for NH_MOEA and STD_MOEA, respectively.

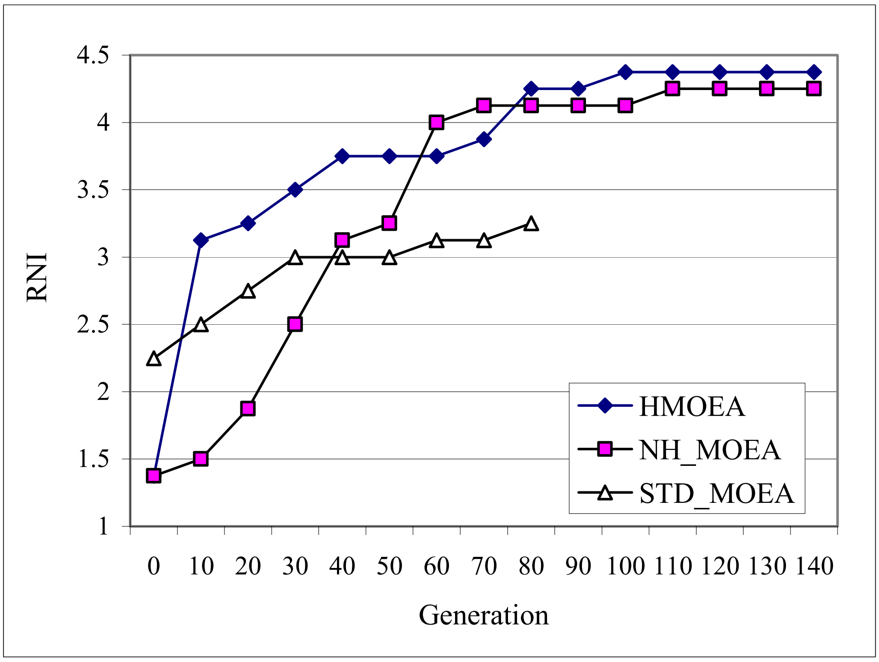


Figure 41 The RNI of various algorithms for test case 132_3_4

5.4.3.3 Simulation Time

Besides the multiobjective optimization performance, the computational time for different algorithms is studied in this sub-section. The three algorithms adopt the same stopping criteria in the simulation, i.e., the evolution stops after 1000 generations or when no improvement is found for the last 5 generations. Fig. 42 shows the normalized simulation time for the three algorithms based on three randomly selected test cases from each category, e.g., test_132_3_4, test_132_tep5 and test_132_ltt5. As can be seen, the STD_MOEA requires the shortest time to converge or halt the evolution, although the optimization results obtained by the

STD_MOEA are much inferior as compared to NH_MOEA and HMOEA. It is believed that the population in STD_MOEA has converged prematurely to local Pareto front. The results also show that the computation time required by HMOEA is better than NH_MOEA for the normal and TEPC categories (which have abundant trailer resources where more feasible solutions exist) and is comparable to NH_MOEA for the LTTC category (which has less trailer resources with a smaller set of feasible solutions).

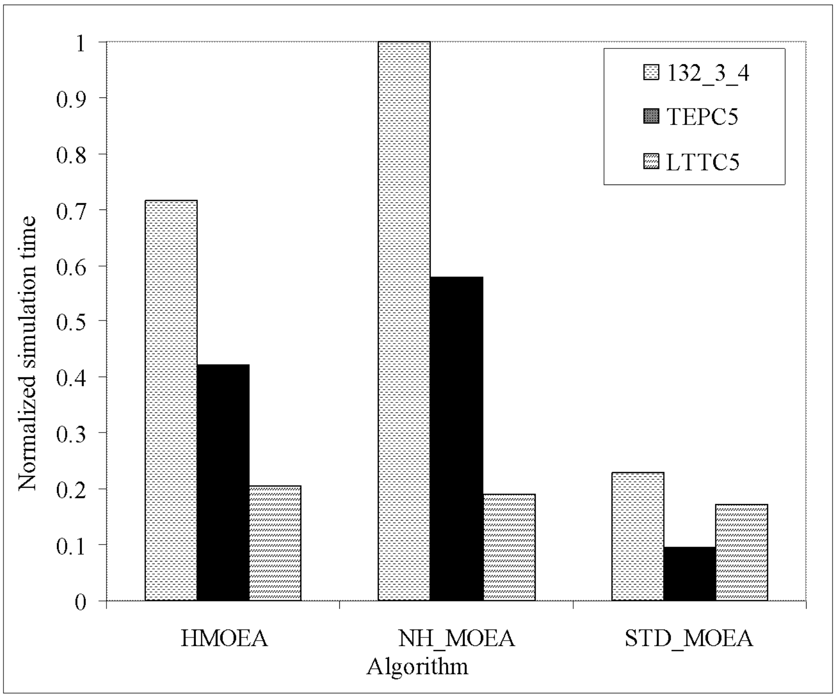


Figure 42 The normalized simulation time for various algorithms

5.5 Conclusion

A transportation problem for moving empty or laden containers for a logistic company has been considered and a mathematical model for the truck and trailer vehicle scheduling problem (TTVSP) has been constructed in this research. The objective of the scheduling problem is to minimize the routing distance and the number of trucks required, subject to a number of constraints such as time windows and availability of trailers. To solve such a multiobjective and multi-modal combinatorial optimization problem, a hybrid multiobjective evolutionary algorithm (HMOEA) featured with specialized genetic operators, variable-length representation and local search heuristic has been applied to find the Pareto optimal scheduling solutions for the TTVSP. Detailed studies have been performed to extract important decision-making information from the multiobjective optimization results. Besides, the relationships among different variables, such as the number of trucks and trailers, the trailer exchange points, and the utilization of trucks in the routing solutions, have been examined and analyzed. The computational results have shown that HMOEA is effective in solving multiobjective combinatorial optimization problems, such as finding useful trade-off solutions for the TTVSP routing problem. Comparisons to two other general evolutionary algorithms also show that the proposed approach is better in terms of the average routing cost and the Ratio of Non-dominated Individuals.

Chapter 6 Conclusions

Unlike many parametric optimization problems, the solution space of the vehicle routing problems is never a clear neighborhood structure, i.e., it is difficult to trace or predict good solutions since feasible solutions may not be located at the neighborhood of current candidate solutions. In addition to that, the real world applications are seldom single objective in nature. Therefore, to provide useful solutions to the decision makers, new approach is required to enhance the existing solutions for multiobjective optimization. The exhaustive review in Chapter 2 examines some of alternatives available. Instances from various applications are categorized to analyze the current landscape of the research domain.

The Vehicle capacity planning system (VCPS) serves as an example of actual real world application that needs optimization for cost reduction purpose. The routing model for container movements is derived from a model in industry problem. In this routing model, outsource is allowed to cover jobs that are not economical if performed internally. Of course, the selection of jobs is not a straightforward process, since many constraints are applicable in getting to the final solution. The results from optimization using new method are proven better than old existing method applied in company. The exposure to such problem has become the motivation to explore better solutions for vehicle routing problem with multiobjective perspective which also subsequently tackled with the proposed HMOEA.

Vehicle routing problem with time windows constraint involves the optimization of routes for multiple vehicles so as to meet all given constraints and to minimize the objectives of travel distance and number of vehicles. A hybrid multiobjective evolutionary algorithm (HMOEA) has been proposed, which incorporates innovative chromosome representation and adapted evolutionary operators to accommodate the sequence-oriented optimization in VRPTW. The HMOEA optimizes very well on VRPTW problems, which improves the routing solutions in many aspects, such as lower routing cost, better population distribution and good convergence trace. Besides, simulations have been performed extensively on the 56 benchmark problems, which yield 20 routing solutions better than or equivalent to the best solutions published in literature.

Following this, TTVSP proposes a new variant of VRP which is similar to VCPS where movement of containers has to be optimized. The model is presented in mathematical modeling together with detail description on the tasks, constraints and objectives. All relevant constraints must be satisfied in every feasible solution. Trailer resource is a critical factor in this problem. The HMOEA is implemented to solve this problem with various test instances. Results from simulation are analyzed and useful information has been extracted from the solutions. The study on trailer resource allocation also provides valuable information to the understanding of the TTVSP problem.

Analyses based on results from benchmark problems show that the performance of HMOEA is consistent. This confirms the reliability of the proposed algorithm, which has shown the robustness to solve problems of varying sizes and difficulties. Through an analysis on how the population had improved over the generations, it demonstrates that the proposed method can drive the population towards Pareto optimality.

The process of optimization can definitely benefit from relevant knowledge and information regarding the problem domain. For example, eliminate any undesired solution space or confine the exploration to smaller search space when certain solution is known to be less desired by the decision maker. The design of evolutionary operators has to be application-aware to improve efficiency in solving different problems. Priori is essential in order to develop effective operators. Without vital information about the problem space, it is not easy to solve a problem optimally. Crossover, mutation and selection operators are the three core substances in the entire optimization algorithm. Therefore, the choice of the operators deserves careful investigation, evaluation and consideration. Through the results, the specialized operators are proven to perform well in solving the vehicle routing and scheduling problems. Nevertheless, the operators should not be over-constrained by priori such that the diversity of a population is not maintained. In general, careful consideration has to be taken when applying priori in any evolutionary optimization algorithm.

In a nutshell, the simplicity of the proposed approach and the elaborated optimization results seem to render it as a promising method for potential future improvements and extensions.

Chapter 7 Future Research

A number of ideas and suggestion were collected as the possibility to enhance and improve the research topic. The results in this thesis lay the groundwork for using HMOEA in solving routing and scheduling applications specifically. Hence, some natural extensions to this work would help to expand and strengthen the results and usages of HMOEA in multiobjective optimization.

7.1 Extensions and improvements

From the standpoint of the algorithm, the evaluation of chromosomes in evolutionary algorithm is an attractive research area to work on for further investigation. The calculation of objectives for a single routing solution is not essentially trivia in the midst of all the timing constraints and costs attached to different routes. The evaluation of chromosomes is taking a substantial amount of computation time. If the evaluation process can be simplified or improved, the run time for simulation can be shortened too. Although the run time is not shown as the most vital factor in current application, a lean and fast algorithm will definitely make it easier to be adopted in some other real world applications.

In fact, the solution can leverage from existing methods in computer science studies, such as an expert system that is able to keep track and store the results of evaluations and avoid the repetition of calculations. Moreover, an innovative classification program, for instance a neural network classifier, can be employed to evaluate the chromosomes. Using specific hardware emulation setup could be another method to speed up the process of evaluation.

Similarly, the advancement in network computing can contribute in improving the capability of MOEA. The computational cost incurred for solving an optimization problem (both time and hardware factors) increases as the size and complexity of the problem escalate. One of the reasons can attribute to the large number of function evaluations in parallel along with the evolution process. Moreover, MOEA in routing and scheduling application usually requires a larger population and generation size in order to simulate the evolutionary model with a better approximation and resolution. The computation load is sometime prohibitive to normal PC users or cannot be performed without the help of high performance computing.

One promising approach to overcome these limitations is to exploit the inherent parallel nature of MOEA by formulating the problem into a distributed computing structure suitable for parallel processing, i.e., to divide a task into subtasks and to solve the subtasks simultaneously using multiple processors. The availability of powerful-networked computers presents a wealth of computing resources that can provide the processing power required to solve those problems. Large problems can be divided into many smaller jobs mapped into the individual computers available in the system. This potential computational power can be much stronger than a supercomputer. Nevertheless, the heterogeneous hardware and software on the Internet has limited the transparency of implementation for distributed systems. Hence, the research to develop an HMOEA that can be used to solve routing and scheduling problem on distributed platform can be a challenging work to tackle all the system transparency issue.

On the other hand, a friendly graphic user interface is required to enhance the user experience with the HMOEA applications. Users are attracted to the solutions that come with an easy and comprehensible interface. The user should be able to relate the problems domain in simple programming language and the HMOEA computation engine that is embedded internally can compute and subsequently propose feasible optimized solutions. Indeed, it is also valuable to display the results of application to user so that user can be convinced that the solutions found are not only optimized but realistic too. To enable the users to visualize the results, a simple but convenient user interface would be one key element.

7.2 Future work

Probing deeper, the results in this thesis also provide a foundation for future work related to vehicle routing and scheduling problems. Diversifying the objectives of optimizations could bring the suggested model one step closer to actual scenario. The combinations of objectives in real world application can be infinite, especially considering the vast variety transportation systems existing in the world. In some developing countries, human labor cost is relatively cheaper and might not impact the routing cost directly. On contrary, some metropolitans may have extremely expensive labor cost that must be measured for every decision-making. Another example could be the logistics of moving chemical products where safety and timely delivery are vital. The biotech companies have many rigid rules and regulations to follow in order to ensure the safety is well taken care of. The objectives also translate to the truth that a new problem model might be required. The fact that HMOEA is extensible to more objectives concurrently, make it a feasible approach to apply even when the decision makers are still evaluating different combinations of cost functions and the associations among these factors.

Nowadays, businesses move fast in order to keep in pace with consumers' need. The business landscape is changing everyday. In many occasions, the stochastic behavior of the consumers' demand is a very common observation. In fact, businesses rely on the capability to react dynamically to any change to survive through tough competition. Such demand hike and slope translate into perturbation to transportation model and is becoming a very challenging problem to the logistic operators. In order to handle stochastic demand in such environment, new model could be devised. Additional constraints might be appended to reflect the dynamism.

The transportation problem model could also be extended to a larger scale. The geographical scope might span to more than one country. Additional locations in the problem can increase the optimization space tremendously. Additionally, the amount of jobs can be increased or the number of trips to depot can be changed accordingly to different customer needs. Research can be performed on how such modification in the problem model would affect the optimizations.

In short, enhancement to the related research can result to an optimization solution with good searching ability (population diversity) that is able to provide near-optimal results and works faster in term of computation time. The domain of vehicle routing and scheduling problem could be extended to the higher scale of complexity with more real world attributes factored into solutions.

Bibliography

Adeli, H. and Cheng, N.T., "Augmented Lagrangian Genetic Algorithm for structural optimization," *Journal of Aerospace Engineering*," vol.7, pp. 104-118, 1994.

Al-Hakim, L., "An analogue genetic algorithm for solving job shop scheduling problem," *International Journal of Production Research*, vol. 39, no. 7, pp. 1537-1548, 2001.

Alexandre, L. B. and Teodor, G.C. "A cooperative parallel meta-heuristic for the vehicle routing problem with time windows," *Computer and Operations Research*, vol. 32, no. 7, pp. 1685-1708, 2005.

Amberg, A., Domschke, W. and Voß, S., "Multiple center capacitated arc routing problems: A tabu search algorithm using capacitated trees," *European Journal of Operational Research*, vol. 124, no. 2, pp. 360-376, 2000.

Applegate, D., Cook, W., Dash, S. and Rohe, A., "Solution of a min-max vehicle routing problem," *INFORMS Journal on Computing*, vol. 14, no. 2, pp. 132-143, 2002.

Aytug, H., Bhattacharyya, S. and Koehler, G., "Genetic learning through simulation: an investigation in shop floor scheduling," *Annals of Operations Research*, vol. 78, pp. 1-29, 1998.

Bäck, T., *Evolutionary Algorithms in Theory and Practice*, New York: Oxford University Press, 1996.

Baek, D. H. and Yoon, W. C., "Co-evolutionary genetic algorithm for multi-machine scheduling: coping with high performance variability," *International Journal of Production Research*, vol. 40, no. 1, pp. 239 – 254, 2002.

Bagchi, T. P., *Multiobjective Scheduling by Genetic Algorithms*, Boston: Kluwer Academic Publishers, 1999.

Baita, F., Pesenti R., Ukovich, W. and Favaretoo, D., "A comparison of different solution approaches to the vehicle scheduling problem in a practical case", *Journal of Computer & Operations Research*, vol. 27, no. 3, pp. 1249-1269, 2000.

Baker, K., *Introduction to Sequencing and Scheduling*, New York: John Wiley, 1974.

Baptista, S., Oliveira, R. C. and Zúquete, E., "A period vehicle routing case study," *European Journal of Operational Research*, vol. 139, no. 2, pp. 220-229, 2002.

Bard, J.F., Kontoravdis, G. and Yu, G., "A branch and cut procedure for the vehicle routing problem with time windows," *Transportation Science*, vol. 36, no. 2, pp. 250-269, 2002.

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

Basu, M., "An interactive fuzzy satisfying method based on evolutionary programming technique for multiobjective short-term hydrothermal scheduling," *Electric Power Systems Research*, vol. 69, no. 2, pp. 277-285, 2004.

Bean, J. C., "Genetics and random keys for sequencing and optimization," *Journal on Computing*, vol. 6, no. 2, pp. 154-160, 1994.

Beasley, J. E. and Christofides, N., "Vehicle routing with a sparse feasibility graph," *European Journal of Operational Research*, vol. 98, no. 3, pp. 499-511, 1997.

Belenguer, J. M., Martinez, M. C. and Mota, E., "A lower bound for the split delivery vehicle routing problem," *Operations Research*, vol. 48, no.5, pp. 801-810, 2000.

Ben, P., Rankin, R. C., Cumming, A. and Fogarty, T. C., "Timetabling the classes of an entire university with an evolutionary algorithm", *Parallel Problem Solving From Nature V, Lecture Notes in Computer Science No. 1498*, A. E. Eiben, T. Back, M. Schoenauer and H. Schwefel, Amsterdam: Springer-Verlag, 1998.

Bent, R. and Van Hentenryck, P., "A two-stage hybrid local search for the vehicle routing problem with time windows," Computer Science Department, Brown University, RI, Technical Report CS-01-06, Sept. 2001.

Berger, J., Barkaoui, M. and Bräysy, O., "A parallel hybrid genetic algorithm for the vehicle routing problem with time windows", Defense Research Establishment Valcartier, Canada, Working Paper, 2001.

Bertsimas, D. and Simchi-Levi, D., "A new generation of vehicle routing research: robust algorithms, addressing uncertainty," *Operations Research*, vol. 44, no. 2, pp. 286-304, 1993.

Beullens, P., Muyldermans, L., Cattrysse, D. and Van Oudheusden, D., "A guided local search heuristic for the capacitated arc routing problem," *European Journal of Operational Research*, vol. 147, no. 3, pp. 629-643, 2003.

Biewirth, C., "A generalized permutation approach to job shop scheduling with genetic algorithms," *OR Spektrum*, vol. 17, pp. 87-92, 1995.

Blanton, J. L. Jr. and Wainwright, R. L., “Multiple vehicles routing with time and capacity constraints using genetic algorithms,” *Fifth International Conference on Genetic Algorithms,* pp. 452-459, 1993.

Boland, N. L., Clarke, L. W. and Nemhauser, G. L., “The asymmetric traveling salesman problem with replenishment arcs,” *European Journal of Operational Research*, vol. 123, no. 2, pp. 408-427, 2000.

Borgulya, I., “A cluster-based evolutionary algorithm for the single machine total weighted tardiness-scheduling problem,” in *Proceedings of the 24th International Conference on Information Technology Interfaces 2002* (ITI 2002), vol. 1, pp. 239-244, 2002.

Brandão, J. and Mercer, A., “A tabu search algorithm for the multi-trip vehicle routing and scheduling problem,” *European Journal of Operational Research*, vol. 100, no. 1, pp. 180-191, 1997.

Bräysy, O., “A reactive variable neighborhood search algorithm for the vehicle routing problem with time windows,” *INFORMS Journal on Computing*, vol. 15, no.4, 2003.

Bräysy, O. and Gendreau M., “Genetic algorithms for the vehicle routing problem with time windows,” SINTEF Applied Mathematics, Department of Optimisation, Oslo, Norway, Internal Report STF42 A01021, 2001a.

Bräysy, O. and Gendreau M., “Tabu search heuristics for the vehicle routing problem with time windows,” SINTEF Applied Mathematics, Department of Optimisation, Oslo, Norway, Internal Report STF42 A01022, 2001b.

Breedam, A. V., “A parametric analysis of heuristics for the vehicle routing problem with side-constraints,” *European Journal of Operational Research*, vol. 137, no. 2, pp. 348-370, 2002.

Breedam, A. V., "Comparing descent heuristic and metaheuristic for the vehicle routing problem," *Computer & Operations Research*, vol. 28, no. 4, pp. 289-315, 2001.

Breedam, A. V., "Improvement heuristics for the vehicle routing problem based on simulated annealing," *European Journal of Operational Research*, vol. 86, no. 3, pp. 480-490, 1995.

Brizuela, C. A. and Sannomiya, N., "Robustness and diversity in genetic algorithms for a complex combinatorial optimization problem," *International Journal of Systems Science*, vol. 32, no. 9, pp. 1161-1168, 2001.

Bruker, P., *Scheduling Algorithms*, 2nd edition, Berlin: Springer-Verlag, 1995.

Bruns, R., "Scheduling," in *Handbook of Evolutionary Computation*, New York: Institute of Physics Publishing and Oxford University Press, pp. F1.5 1-9, 1997.

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

Burke, E., Cowling, P., De Causmaecker, P., Vanden Berghe, G., "A memetic approach to the nurse rostering problem," *Applied Intelligence*, vol. 15, no. 3, pp. 199-214, 2001a.

Burke, E. K., De Causmaecker, P., Petrovic, S. and Berghe, G. V., "Fitness evaluation for nurse scheduling problems," in *Proceedings of the Congress on Evolutionary Computation 2001*, vol. 2, pp. 1139-1146, 2001b.

Burke, E. K. and Petrovic, S., “Recent research directions in automated timetabling,” *European Journal of Operational Research*, vol. 140, pp. 266-280, 2002.

Burke, E. K. and Smith, A. J., “Hybrid evolutionary techniques for the maintenance scheduling problem,” *IEEE Transactions on Power Systems*, vol. 15, no. 1, pp. 122-128, 2000.

Burke, E. K. and Newall, J. P., “A multi-stage evolutionary algorithm for the timetable problem,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 1, pp. 63-74, 1999.

Caseau, Y. and Laburthe, F., “Heuristics for large constrained vehicle routing problems,” *Journal of Heuristics*, vol. 5, no. 3, pp. 281-303, 1999.

Cavalieri, S. and Gaiardelli, P., “Hybrid genetic algorithms for a multiple-objective scheduling problem,” *Journal of Intelligent Manufacturing*, vol. 9, no. 4, pp. 361-367, 1997

Celano, G., Costa, A., Fichera, S. and Perrone, G., “Fuzzy scheduling of a flexible assembly line through an evolutionary algorithm,” in *IEEE International Conference on Systems, Man, and Cybernetics 2000*, vol. 1, pp. 328-333, 2000.

Chambers, J. M., Cleveland, W. S., Kleiner, B., and Turkey, P. A. *Graphical Methods for Data Analysis*, Belmont, CA: Wadsworth, 1983.

Chan, W. T. and Hu, H., “Precast production scheduling with genetic algorithms,” in *Proceedings of the Congress on Evolutionary Computation 2000*, vol. 2, pp. 1087-1094, 2000.

Chao, I-Ming, “A tabu search method for the truck and trailer routing problem”, *Computers And Operations Research*, vol. 29, no.1, pp. 33-51, 2002.

Chavalitwongse, W., Kim, D. and Pardalos, P.M., "GRASP with a new local search scheme for vehicle routing problems with time windows," *Journal of Combinatorial Optimization*, vol. 7, no. 2, pp. 179-207, 2003.

Chen C. L., Neppalli, R. V. and Aljabel, N. "Genetic algorithms applied to the continuous flowshop problem, *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 919-929, 1996.

Cheung, W. and Zhou, H., "Using genetic algorithms and heuristics for job shop scheduling with sequence-dependent setup times," *Annals of Operations Research*, vol. 107, no. 1, pp. 65-81, 2001.

Chiang, W. C. and Russel, R. A., "Simulated annealing metaheuristic for the vehicle routing problem with time windows," *Annals of Operations Research*, vol. 63, no. 1, pp. 3-27, 1996.

Chiang, W. C. and Russel, R. A., "A reactive tabu search metaheuristics for the vehicle routing problem with time windows", *INFORMS Journal on Computing*, vol. 9, pp. 417-430, 1997.

Christofides, N., Mingozzi, A., Toth, P., "Exact algorithms for the vehicle routing problem based on spanning tree and shortest path relaxations," *Math. Programming,* vol. 20, no.3, pp. 255-282, 1981.

Chryssolouris, G. and Subramaniam, V., "Dynamic scheduling of manufacturing job shops using genetic algorithms", *Journal of Intelligent Manufacturing*, vol. 12, no. 3, pp. 281-293, 2001.

Chung, Y. L., Lei, L. and Pinedo, M., "Current trends in deterministic scheduling," *Annals of Operations Research*, vol. 70, pp. 1-41, 1997.

Coello Coello, C. A., “A comprehensive survey of evolutionary-based multiobjective optimization techniques”, *Knowledge and Information Systems*, vol. 1, no. 3, pp. 269-308, 1999.

Coello Coello, C.A. “An updated survey of GA-based multiobjective optimization techniques,” Laboratorio Nacional de Informatica Avanzada (LANIA), Xalapa, Veracruz, Mexico, Report No. Lania-RD-98-08, 1998.

Coello Coello, C. A., Van Veldhuizen, D. A., Lamont, G. B., *Evolutionary Algorithms for Solving Multi-Objective Problems*, Kluwer Academic/Plenum Publishers, 2002.

Cook, W. and Rich, J. L., “A parallel cutting plan algorithm for the vehicle routing problem with time windows,” Computational and Applied Mathematics Department, Rice University, Houston, TX, Technical Report, 1999.

Cordeau, J. F., Gendreau, M., Laporte, G., Potvin, J. Y. and Semet, F., “A guide to vehicle routing heuristics”, Journal of the Operational Research Society, vol. 53, no. 5, pp.512-522, 2002.

Cordeau, J. F., Laporte, G., and Mercier, A., "A unified tabu search heuristic for vehicle routing problems with time windows," *Journal of the Operational Research Society*, vol. 52, no. 8, pp. 928-936, 2001.

Cordone, R. and Wolfler-Calvo, R., “A heuristic for the vehicle routing problem with time windows,” *Journal of Heuristics*, vol. 7, no. 2, pp. 107-129, 2001.

Costa, A., Celano, G. and Fichera, S., “Fuzzy scheduling of a flexible assembly line through evolutionary algorithms,” *IEEE International Conference on Systems, Man, and Cybernetics 2001*, vol. 4, pp. 2251-2256, 2001.

Cowling, P., Kendall, G. and Han, L., “An investigation of a hyperheuristic genetic algorithm applied to a trainer scheduling problem,” in *Proceedings of the*

Congress on Evolutionary Computation 2002 (CEC 02), vol. 2, pp. 1185-1190, 2002.

Cvetkovic, D., and Parmee I. C., "Preferences and their application in evolutionary multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 42-57, 2002.

Czech, Z. J. and Czarnas, P., "A parallel simulated annealing for the vehicle routing problem with time windows," *Proc. 10th Euromicro Workshop on Parallel, Distributed and Network-based Processing*, Canary Islands, Spain, pp. 376-383, 2002.

Dahal, K. P., Burt, G. M., McDonald, J. R. and Moyes, A., "A case study of scheduling storage tanks using a hybrid genetic algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 5 , no. 3, pp. 283-294, 2001.

Dahal, K. P., Burt, G. M., McDonald, J. R. and Galloway, S. J., "GA/SA-based hybrid techniques for the scheduling of generator maintenance in power systems," in *Proceedings of the Congress on Evolutionary Computation 2000*, vol. 1, pp. 567-574, 2000.

Darwen, P. J., "Search landscape of a realistic single-machine scheduling task: peaks with big differences," in *Proceedings of the Congress on Evolutionary Computation 2002* (CEC 2002), vol. 2, pp. 1191-1196, 2002.

Davis L., "Job shop scheduling with genetic algorithms," In Grefenstette, J. J. (editor), in *Proceeding of the International Conference on Genetic Algorithms and their Applications*, San Mateo: Morgan Kaufmann, pp. 136-140, 1985.

De Almeida, M. R., Hamacher, S., Pacheco, M. A. C. and Vellasco, M. B. R., "The energy minimization method: a multiobjective fitness evaluation technique and its application to the production scheduling in a petroleum refinery," in *Proceedings of the Congress on Evolutionary Computation 2001*, vol. 1, pp. 560-567, 2001.

De Backer, B. and Furnon, V., Kilby P., Prosser P. and Shaw P., "Solving vehicle routing problems using constraint programming and metaheuristics," *Journal of Heuristics*, vol. 6, no. 4, pp. 501-523, 2002.

Deb, K. and Tiwari, S., "Multi-objective optimization of a leg mechanism using genetic algorithms," *Engineering Optimization*, vol. 37, no. 4, pp. 325-350, 2005.

Deb, K., Jain, P., Gupta, N. K. and Maji, H.K., "Multiobjective placement of electronic components using evolutionary algorithms", *IEEE Transactions on Components and Packaging Technologies, Part A: Packaging Technologies*, vol. 27, no. 3, pp. 480-492, 2004.

Deb, K., "Unveiling innovative design principles by means of multiple conflicting objectives," *Engineering Optimization*, vol. 35, no. 5, pp. 445-470, 2003.

Deb, K., Reddy, A. R. and Singh, G., "Optimal scheduling of casting sequence using genetic algorithms," *Journal of Materials and Manufacturing Processes*, vol.18, no. 3, pp. 409-432, 2003.

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

Deb, K., Pratap. A, Agarwal, S. and Meyarivan, T., "A fast and elitist multi-objective genetic algorithm: NSGA-II," *IEEE Transaction on Evolutionary Computation*, vol. 6, no. 2, pp. 181-197, 2002b.

Deb, K., *Multi-objective Optimization using Evolutionary Algorithms*, London: John Wiley & Sons, 2001a.

Deb, K., “Nonlinear goal programming using multi-objective genetic algorithms,” *Journal of the Operational Research Society*, vol. 52, no. 3, pp. 291-302, 2001b.

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

Deb, K., “An efficient constraint handling method for genetic algorithms,” *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2-4, pp. 311-338, 2000.

Deb, K., “Multi-objective genetic algorithms: Problem difficulties and construction of test problems,” *Evolutionary Computation Journal*, vol. 7, no. 3, pp. 205-230, 1999.

Deb, K. and Chakroborty, P., “Time scheduling of transit systems with transfer considerations using genetic algorithms,” *Evolutionary Computation Journal*, vol. 6, no. 1, pp. 1-24, 1998.

Desrochers, M., Desrosiers, J. and Solomon, M., “A new optimization algorithm for the vehicle routing problem with time windows,” *Operational Research*, vol. 40, no. 2, pp. 342-354, 1992.

Desrosier, J., Dumas, Y., Solomon, M., and Soumis, F., “Time constraint routing and scheduling”, *Handbooks in Operations Research and Management Science 8*: *Network Routing*, M. Ball (eds.), Amsterdam: Elsevier Science Publishers, pp. 35-139, 1995.

Dhodhi, M. K., Ahmad, I., Yatama, A. and Ahmad, I., “An integrated technique for task matching and scheduling onto distributed heterogeneous computing systems,” *Journal of Parallel and Distributed Computing*, vol. 62, no. 9, pp. 1338-1361, 2002.

Dias, A. H. F. and Vasconcelos, J. A., “Multiobjective genetic algorithms applied to solve optimization problems”, *IEEE Transactions on Magnetic*, vol. 38, no. 2, pp. 1133-1136, 2002.

Dijkstra, E. W, “A note on two problems in connexion with graphs,” *Numerische Mathematick*, vol. 1, pp. 269-271, 1959.

Dillmann, R., Becker, B. and Beckefeld, V., “Practical aspects of route planning for magazine and newspaper wholesalers,” *European Journal of Operational Research*, vol. 90, no. 1, pp. 1-12, 1996.

Dimopoulos, C. and Zalzala, A. M. S., “Investigating the use of genetic programming for a classic one-machine scheduling problem,” Advances in Engineering Software, vol. 32, no. 6, pp. 489-498, 2001.

Dimopoulos, C. and Zalzala, A. M. S., “Recent developments in Evolutionary computation for manufacturing optimization: problems, solutions, and comparisons,” *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 2, pp. 93-113, 2000.

Doerner, K., Gronalt, M., Hartl, R. F., Reimann, M. and Zisser, K., “VRP with interdependent time windows - A case study for the Austrian Red Cross blood programme,” in *Operations Research Proceedings* 2002, Berlin/Heidelberg: Springer, pp. 156-161, 2002.

Dorndorf, U. and Pesch, E., “Evolution based learning in a job shop scheduling environment,” *Computers and Operations Research*, vol. 22, pp. 25-40, 1995.

Dror, M., Arc Routing: Theory, Solutions, And Applications, Boston, MA : Kluwer Academic, 2000.

Dullaert, W., Janssens, G. K., Sörensen, K. and Vernimmen, B., "New heuristics for the fleet size and mix vehicle routing problem with time windows," *Journal of the Operational Research Society*, vol. 53, no. 11, pp. 1232-1238, 2002.

Dullaert, W., "Impact of relative route length on the choice of time insertion criteria for insertion heuristics for the vehicle routing problem with time windows", in *The Rome Jubilee 2000 Conference Improving Knowledge and Tools for Transportation and Logistics Development: 8th Meeting of the Euro Working Group Transportation*, Maurizio, B. (ed.), pp. 153-156, 2000.

Elrad, T. and Lin, J., "Evolving processes and evolution schedulers for concurrent scheduling controls and parallel evolutionary computation," *Future Generation Computer Systems*, vol. 14, no. 5-6, pp. 365-381, 1998.

El-Sharkh, M. Y. and El-Keib, A. A., "An evolutionary programming-based solution methodology for power generation and transmission maintenance scheduling," *Electric Power Systems Research*, vol. 65, no. 1, pp. 35-40, 2003a.

El-Sharkh, M. Y. and El-Keib, A. A., "Maintenance scheduling of generation and transmission systems using fuzzy evolutionary programming," *IEEE Transactions on Power Systems*, vol. 18, no. 2, pp. 862-866, 2003b.

El-Sharkh, M. Y., El-Keib, A. A. and Chen, H., "A fuzzy evolutionary programming-based solution methodology for security-constrained generation maintenance scheduling," *Electric Power Systems Research*, vol. 67, no. 1, pp. 67-72, 2003c.

Esquivel, S., Ferrero, S., Gallard, R., Salto, C., Alfonso, H. and Schütz, M., "Enhanced evolutionary algorithms for single and multiobjective optimization in the job shop scheduling problem," *Knowledge-Based Systems*, vol. 15, no. 1-2, pp. 13-25, 2002.

Fang, H. L., Ross, P. and Corne, D, "A promising genetic algorithm approach to job shop scheduling, rescheduling, and open-shop scheduling problems," in *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 375-382, 1993.

Fayech, B., Hammadi, S. and Maouche, S., "Partial network reconfiguration through a dynamic vehicle scheduling approach," *IEEE International Conference on Systems, Man and Cybernetics 2002*, vol. 4, 2002.

Feldmann, M. and Biskup, D., "Single-machine scheduling for minimizing earliness and tardiness penalties by meta-heuristic approaches," *Computers & Industrial Engineering*, vol. 44, no. 2, pp. 307-323, 2003.

Fisher, M. L., "Vehicle routing", in *Handbooks in Operations Research and Management Science 8*: *Network Routing*, M. Ball, (eds), Amsterdam: Elsevier Science Publishers, pp. 1-33, 1995.

Fölsz, F., Mészáros, C. and Rapcsák, T., "Distribution of gas cylinders," *European Journal of Operational Research*, vol. 87, no. 3, pp. 613-623, 1995.

Fogel, D. B. and Fogel, L. J., "Using evolutionary programming to schedule tasks on a suite of heterogeneous computers," *Computer Operations Research*, vol. 23, no. 6, pp. 527- 534, 1996.

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

Fonseca, C. M. *Multiobjective Genetic Algorithms with Application to Control Engineering Problems*, Dept. Automatic Control and Systems Eng., University of Sheffield, Sheffield, UK, Ph.D. Thesis, 1995.

Fonseca, C. M. and Fleming, P. J. "Genetic algorithm for multiobjective optimization, formulation, discussion and generalization", in *Genetic Algorithms: Proceeding of the Fifth International Conference.* Morgan Kaufmann, San Mateo, CA, pp. 416-423, 1993.

Fu, Michael C., "Optimization for simulation: theory vs. Practice," *INFORMS Journal on Computing*, vol. 14, no. 3, pp. 192-215, 2002.

Gambardella, L. M., Taillard, E., Agazzi, G., "MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows," In *New Ideas in Optimization*, Corne D. (eds.). McGraw-Hill, 1999.

Garey, M. R. and Johnson, D. S, *Computers and Intractability: a Guide to the Theory of NP-Completeness*, San Francisco: Freeman, 1979.

Garey, M. R., Johnson, D. S. and Sethi, R., "The complexity of flowshop and jobshop scheduling," *Mathematics of Operations Research*, vol. 1, pp. 117-129, 1976.

Gehring, H. and Homberger, J, "A parallel two phase metaheuristic for routing problems with time windows," *Asia-Pacific Journal of Operation Research,* vol. 18, no.1, pp. 35-47, 2001.

Gehring H. and Homberger, J, "Parallelization of a two-phase metaheuristic for routing problems with time windows," *Journal of Heuristics*, vol. 3, no. 8, pp. 251-276, 2002.

Gendreau, M., Laporte, G. and Potvin, J.Y., "Metaheuristics for the vehicle routing problem," University of Montreal, Canada, Les Cahiers du GERAD G-98-52, 1999.

Gerdeseen, J. C. “Vehicle Routing Problem with Trailers”, *European Journal of Operational Research*, vol. 93, no. 1, pp. 135-137, 1996.

Gendreau, M., Laporte, G., Musaraganyi, C. and Taillard, E. D., “A tabu search heuristic for heterogeneous fleet vehicle routing problem,” *Computers and Operations Research*, vol. 26, no. 12, pp. 1153-1173, 1999a.

Gendreau, M., Laporte, G. and Potvin, J.Y., “Metaheuristics for the vehicle routing problem,” University of Montreal, Canada, Les Cahiers du GERAD G-98-52, 1999b.

Gendreau, M., Guertin, F., Potvin, J. Y. and Taillard, E. D., “Parallel tabu search for real-time vehicle routing and dispatching,” *Transportation Science*, vol. 33, no.4, pp. 381-390, 1999c.

Gezdur, A. and Türkay, M., “MILP solution to the vehicle routing problem with time windows and discrete vehicle capacities”, in *XXIII National Operations Research and Industrial Engineering Congress*, Istanbul, Turkey, July 2002.

Ghiani, G. and Improta, G.,“An efficient transformation of the generalized vehicle routing problem,” *European Journal of Operational Research*, vol. 122, no. 1, pp. 11-17, 2000.

Glaab, H., “A new variant of a vehicle routing problem: Lower and upper bounds,” *European Journal of Operational Research*, vol. 139, no. 3, pp. 557-577, 2002.

Glibovets, N. N. and Medvid, S. A., “Genetic algorithms used to solve scheduling problems,” *Cybernetics and Systems Analysis*, vol. 39, no. 1, pp. 81-90, 2003.

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

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

Goldberg, D. E., and Lingle, R., "Alleles, loci and the traveling salesman problem," *First International Conference on Genetic Algorithms*, Lawrence Erlbaum, Hillsdale, NJ, pp. 154-159, 1985.

Golden, B. L. and Assad, A. A., *Vehicle Routing: Methods and Studies*, North-Holland, Amsterdam, 1988.

Grefenstette, J. J., Gopal, R., Rosmaita, B. and Van Gucht, D., "Genetic algorithms for the traveling salesman problem," in *Proc. First Intl. Conf. Genetic Algorithms and Their Applications*, pp. 160-168, 1985.

Gürsel A. Süer, Ramon Vazquez and Joel Santos, "Evolutionary programming for minimizing the average flow time in the presence of non-zero ready times," *Computers & Industrial Engineering*, vol. 45, no. 2, pp. 331-344, 2003.

Handa, H., Chapman, L. and Yao, X., "Dynamic Salting Route Optimisation using Evolutionary Computation", *IEEE Congress on Evolutionary Computation*, vol. 1, pp. 158-165, 2005.

Hart, E. and Ross, P., "GAVEL - a new tool for genetic algorithm visualization," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 4, pp. 335-348, 2001.

Hart, E., Ross, P. M. and Nelson, J., "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.

Hartmann, S., "Project scheduling with multiple modes: a genetic algorithm," *Annals of Operations Research*, vol. 102, no. 1, pp. 111-135, 2001.

Hertz, A. and Mittaz, M., "A variable neighbourhood descent algorithm for the undirected capacitated arc routing problem," *Transportation Science*, vol. 35, no. 4, pp. 425-434, 2001.

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

Ho, W. K., Chin, H. J. and Lim, A., "A hybrid search algorithm for the vehicle routing problem with time windows," *International Journal on Artificial Intelligence Tools*, vol. 10, pp. 431-449, 2001.

Holland, J. H., *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, MI, 1975.

Holland, J. H., "Genetic algorithms and the optimal allocation of trials," *SIAM Journal on Computing*, vol. 2, pp. 88-105, 1973.

Homberger, J. and Gehring, H., "Two evolutionary metaheuristic for the vehicle routing problem with time windows," *INFOR*, vol. 37, no. 1, pp. 297-318, 1999.

Horn, J., N. Nafpliotis and D.E. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," in *IEEE International Conference on Evolutionary Computation*, 1994, vol.1, pp. 82-87.

Hsu, T., Dupas, R. and Goncalves, G., "A genetic algorithm to solving the problem of FMS cyclic scheduling," *IEEE Systems, Man and Cybernetics (SMC02)*, Hammamet, Tunisia, 2002.

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

Ichoua, S., Gendreau, M. and Potvin, J. Y., “Vehicle dispatching with time-dependent travel times,” *European Journal of Operational Research*, vol. 144, no. 2, pp. 379-396, 2003.

Inoue, T., Furuhashi, T., Maeda, H. and Takaba, M., “A proposal of combined method of evolutionary algorithm and heuristics for nurse scheduling support system,” *IEEE Transactions on Industrial Electronics*, vol. 50, no. 5, pp. 833-838, 2003.

Ioannou G., Kritikos, M. and Prastacos, G., “A greedy look ahead heuristic for the vehicle routing problem with time windows,” *Journal of the Operational Research Society*, vol. 52, no. 5, pp. 523-537, 2001.

Ishibashi, H., Aguirre, H., Tanaka, K. and Sugimura, T., "Multi-objective optimization with improved genetic algorithm”, *IEEE International Conference on Systems, Man, and Cybernetics (SMC2000)*, Nashville, pp. 3852-3857, 2000.

Ishibuchi, H., Yoshida, T. and Murata, T., “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.

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

Jakob, W., M., Gorges-Schleuter and C. Blume, “Application of genetic algorithms to job planning and learning in parallel problem solving from nature,” 2nd Workshop, ed. by R. Männer and B. Nanderick, *Lecture Notes in Computer Science*, North-Holland Publishing Company, Amsterdam, pp. 291-300. 1992.

Jan, A., Yamamoto, M. and Ohuchi, A., "Evolutionary algorithms for nurse scheduling problem," in *Proceedings of the Congress on Evolutionary Computation 2000*, vol. 1, pp. 196-203, 2000.

Jaszkiewicz, A., "Multiple objective metaheuristic algorithms for combinatorial optimization," Poznań University of Technology, Poznan, Habilitation thesis 360, 2001.

Jaszkiewicz, A., "Genetic local search for multiple objective combinatorial optimization," Institute of Computing Science, Poznań University of Technology, Research Report RA-014/98, 1998.

Jia, H. Z., Nee, A. Y. C., Fuh, J. Y. H. and Zhang, Y. F., "A modified genetic algorithm for distributed scheduling problems," *Journal of Intelligent Manufacturing*, vol. 14, no. 3, pp. 351-362, 2003.

Jozefowiez, N., Semet, F. and Talbi, E., "Parallel and hybrid models for multi-objective optimization: application to the vehicle routing problem," *Parallel Problem Solving from Nature, Lecture Notes in Computer Science*, New York: Springer-Verlag, pp. 271-282, 2002.

Jung, S. and Moon, B. R., "A hybrid genetic algorithm for the vehicle routing problem with time windows," *The Genetic And Evolutionary Computation Conference,* San Francisco: Morgan Kaufmann publishers, pp 1309-1316, 2002.

Kacem, I., Hammadi, S. and Borne, P., "Pareto-optimality approach for flexible job-shop scheduling problems: hybridization of evolutionary algorithms and fuzzy logic," *Journal of Mathematics and Computers in Simulation*, vol. 60, no. 3-5, pp. 245-276, 2002a.

Kacem, I.; Hammadi, S. and Borne, P., "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, 2002b.

Kammarti, R. Hammadi, S., Borne, P. and Ksouri, M., "Lower bounds In an hybrid evolutionary approach for the pickup and delivery problem with time windows," *IEEE International Conference on Systems, Man and Cybernetics*, vol.2, pp. 1156-1161, 2005.

Kallehauge, B., Larsen, J. and Madsen, O.B.G.. "Lagrangean duality applied on vehicle routing with time windows," IMM, Technical University of Denmark, Technical Report IMM-TR-2001-9, 2001.

Karkazis, J. and Boffey, T. B., "Optimal location of routes for vehicles transporting hazardous materials," *European Journal of Operational Research*, vol. 86, no. 2, pp. 201-215, 1995.

Käschel, J., Teich, T. and Zacher, b., "Real-time dynamic shop floor scheduling using Evolutionary Algorithms," *International Journal of Production Economics*, vol. 79, no. 2, pp. 113-120, 2002.

Kawanaka, H., Yamamoto, K., Yoshikawa, T., Shinogi, T. and Tsuruoka, S., "Genetic algorithm with the constraints for nurse scheduling problem," in *Proceedings of the Congress on Evolutionary Computation 2001*, vol. 2, pp. 1123-1130, 2001.

Kelly, J. P. and Xu, J., "A Set-Partitioning-Based Heuristic for the Vehicle Routing Problem," *INFORMS Journal on Computing*, vol. 11, no. 2, pp. 161-172, 1999.

Kelly, James P., "Simulation optimization is evolving," INFORMS, *Journal on Computing*, vol. 14, no. 3, pp. 223-225, 2002.

Kenyon, A. S. and Morton, D. P., “Stochastic vehicle routing with random travel times,” *Transportation Science*, vol. 37, no. 1, pp. 69-82, 2003.

Keung, K. W., Ip, W. H. and Lee, T. C., “A genetic algorithm approach to the multiple machine tool selection problem,” *Journal of Intelligent Manufacturing*, vol. 12, no.4, pp. 331-342, 2001.

Kim, H. and Ahn, B., “A new evolutionary algorithm based on sheep flocks heredity model,” *IEEE Pacific Rim Conference on Communications, Computers and Signal Processing 2001* (PACRIM 2001), vol. 2, pp. 514-517, 2001.

Kilby, P. J., Prosser, P. and Shaw, P., “A comparison of traditional and constraint-based heuristic methods on vehicle routing problems with side constraints,” *Journal of Constraints*, vol. 5, no. 4, pp. 389-414, 2000.

Kilby, P. J., Prosser, P. and Shaw, P., “Guided local search for the vehicle routing problem with time windows", in *Meta Heuristics: Advances And Trends In Local Search Paradigms For Optimisation*, Kluwer Academic Publishers, pp. 473-486, 1999.

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

Kohl, N., Desrosiers J., Madsen, O. B. G., Solomon, M. M. and Soumis, F., “2 path cuts for the vehicle routing problem with time windows,” *Transportation Science*, vol. 33, no. 1, pp. 101-116, 1999.

Kohlmorgen, U., Schmeck, H. and Haase K., “Experiences with fine-grained parallel genetic algorithms,” *Annals of Operations Research*, vol. 90, pp. 203-219, 1999.

Kwan, R. S. K. and Mistry, P., "A co-evolutionary algorithm for train timetabling," in *Proceedings of the Congress on Evolutionary Computation 2003* (CEC 2003), vol. 3, pp. 2142-2148, 2003.

Kwan, R. S. K., Wren, A. and Kwan, A. S. K., "Hybrid genetic algorithms for scheduling bus and train drivers," *Proceedings of the Congress on Evolutionary Computation 2000*, vol. 1, pp. 285-292, 2000.

Laporte, G., Louveaux, F. V., and Hamme, L. V., "An integer L-shaped algorithm for the capacitated vehicle routing problem with stochastic demands," *Operations Research*, vol. 50, no. 3, pp. 415-423, 2002.

Laporte, G., "The vehicle routing problem: an overview of exact and approximate algorithms," *Europe Journal of Operational Research*, vol. 59, no.3, pp. 345-358, 1992.

Laporte, G., Gendreau, M. and Potvin J. Y., Semet, F., "Classical and modern heuristics for the vehicle routing problem", *International Transaction in Operational Research*, vol. 7, pp. 285-300, 2000.

Lau, H. C., Lim, Y. F. and Liu, Q. Z., "Diversification of Search Neighborhood via Constraint-Based Local Search and Its Applications to VRPTW," *3rd International Workshop on Integration of AI and OR Techniques (CP-AI-OR)*, Kent, United Kingdom, pp. 1-15, 2001.

Lau, H. C., Sim, M. and Teo, K. M., "Vehicle routing problem with time windows and a limited number of vehicles," *European Journal of Operational Research*, vol. 148, no. 3, pp. 559 –569, 2003.

Lee, L. H., Tan, K. C., Ou, K. and Chew, Y. H., "Vehicle Capacity Planning System (VCPS): A case study on vehicle routing problem with time windows," *IEEE*

Transactions on Systems, Man and Cybernetics: Part A (Systems and Humans), vol. 33, no. 2, pp. 169-178, 2003.

Lee, J. B. and Jeong, J. H., "A daily optimal operational schedule for cogeneration systems in a paper mill," *IEEE Power Engineering Society Summer Meeting 2001*, vol. 3, pp. 1357-1362, 2001.

Li, H. and Lim, A., "Local search with annealing-like restarts to solve the vehicle routing problem with time windows," *ACM Symposium on Applied Computing (SAC 2002)*, pp. 560-565, 2002.

Li, J. and Kwan, R.S.K., "A fuzzy simulated evolution algorithm for the driver scheduling problem," in *Proceedings of the 2001 Congress on Evolutionary Computation, 2001*, vol. 2, pp. 1115-1122, 2001.

Li, Y., Ma, K. F., Tang, K. S. and IP, W. H., "Genetic algorithm to production planning and scheduling problem manufacturing system," *Production Planning and Control*, vol. 11, no. 5, pp. 443-458, 2000.

Lim and Fan, W., "Multi-depot vehicle routing problem: a one-stage approach", *IEEE Transactions on Automation Science and Engineering*, vol. 2, no. 4, pp. 397-402, 2005.

Lim, A. and Zhang, X., "A two-stage heuristic for the vehicle routing problem with time windows and a limited number of vehicles," *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, pp. 82-90, 2005.

Lin, S., "Computer Solutions for Traveling Salesman Problem," *Bell System Technical Journal*, vol. 44, pp. 2245-2269, 1965.

Liu, F. F. and Shen, S. Y., “A route-neighborhood-based metaheuristic for vehicle routing problem with time windows,” *European Journal of Operational Research*, vol. 118, no. 3, pp. 485-504, 1999.

Liu, M. and Wu, C., “Scheduling algorithm based on evolutionary computing in identical parallel machine production line,” *Robotics and Computer-Integrated Manufacturing*, vol. 19, no. 5, pp. 401-407, 2003.

Louis, S. J., Yin, X. and Yuan, Z. Y., “Multiple vehicle routing with time windows using genetic algorithms,” *Proceedings of the Congress on Evolutionary Computation*, pp. 1804-1808, 1999.

Lourenço, H. R., Paixão, J. P. and Portugal, R., “Multiobjective metaheuristics for the bus driver scheduling problem,” *Transporatoin Science*, vol. 35, no. 3, pp. 331- 343, 2001.

Luo, Y-C and Guignard-Spielberg, M., “A hybrid approach for integer programming combining genetic algorithms, linear programming, and ordinal programming,” *Journal of Intelligent Manufacturing*, vol. 12, no. 5, pp. 509-519, 2001.

Madureira, A., Ramos, C. and do Carmo Silva, S. D., “A coordination mechanism for real world scheduling problems using genetic algorithms,” in *Proceedings of the Congress on Evolutionary Computation 2002*, (CEC 2002), vol. 1, pp. 175-180, 2002.

Malmborg, C. J., “A genetic algorithm for service level based vehicle scheduling,” *European Journal of Operational Research*, vol. 93, no. 1, pp. 121-134, 1996.

Manzanedo, F., Donsion, M. P. and Castro, J. L., “An evolution based algorithm for environmentally constrained thermal scheduling problems,” *IEEE PowerTech Conference 2001*, vol. 2, 2001.

Maritime and Port Authority of Singapore, (2002, Nov.). *Singapore Maritime Portal*, [Online]. Available: http://www.singaporemaritimeportal.com/imc_globalport.htm

Michalewicz, Z., Fogel, D. B. and Michaelewica, A., *How to Solve It: Modern Heuristics*, Springer-Verlag, 1999.

Michalewicz, Z., Dasgupta, D., Le Riche, R. G. and Schoenauer, M., "Evolutionary algorithms for constrained engineering problems," *Computers & Industrial Engineering*, vol.30, no.2, pp. 851-870, 1996.

Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution programs*, 2nd edn., New York: Springer-Verlag, 1994.

Middendorf, M., Scheuermann, B., Schmeck, H. and ElGindy, H., "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.

Miller, D. M., Chen, H. C., Matson, J. O. and Liu, Q., "A hybrid genetic algorithm for the single machine scheduling problem," *Journal of Heuristics*, vol. 5, no. 4, pp. 437-454, 1999.

Min, H., Jayaraman, N. and Srivastava, R., "Combined location-routing problems: A synthesis and future research directions," *European Journal of Operational Research*, vol. 108, no. 1, pp. 1-15, 1998.

Mingozzi, A., Giorgi, S. and Baldacci, R. "An exact method for the vehicle routing problem with backhauls," *Transportation Science*, vol. 33, no. 3, pp. 315-329, 1999.

Miwa, M., Inoue, T., Matsuzaki, M., Furuhasi, T. and Okuwa, S., "Nurse scheduling system using bacterial evolutionary algorithm hardware," in *28th Annual*

Conference of the Industrial Electronics Society 2002 (IECON 02), vol. 3, pp. 1801-1805, 2002.

Morad, N. and Zalzala, A., "Genetic algorithms in integrated process planning and scheduling," *Journal of Intelligent Manufacturing*, vol. 10, no. 2, pp. 169-179, 1999.

Mourão, M. C. and Almeida, M. T., "Lower-bounding and heuristic methods for a refuse collection vehicle routing problem," *European Journal of Operational Research*, vol. 121, no. 2, pp. 420-434, 2000.

Murata T., Ishibuchi, H., "Performance evaluation of genetic algorithms for flow shop scheduling problems," *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 1061-1071, 1996.

Murata, T., Ishibuchi, H. and Tanaka, H., "Multi-objective genetic algorithm and its applications to flowshop scheduling," *Computers and Industrial Engineering*, vol. 30, no. 4, pp. 957-968, 1996.

Muyldermans, L., Cattrysse, D., Van Oudheusden, and Lotan, T., "Districting for salt spreading operations," *European Journal of Operational Research*, vol. 139, no. 3, pp. 521-532, 2002.

Nidul Sinha, Chakrabarti, R. and Chattopadhyay, P. K., "Fast evolutionary programming techniques for short-term hydrothermal scheduling," *Electric Power Systems Research*, vol. 66, no. 2, pp. 97-103, 2003.

Norman, B. A. and Bean, J. C., "Scheduling operations on parallel machine tools," *IIE Transactions*, vol. 32, no. 5, pp. 449-460, 2000.

Oliver, I. M., Smith, D. J. and Holland, J. R. C., "A study of permutation crossover operators on the traveling salesman problem," in *Proceedings of The Second ICGA*, New Jersey: Lawrence Erlbaum Associates, pp. 224-230, 1987.

Osman I. H. and Christofides N., "Simulated annealing and descent algorithms for capacitated clustering problem", Imperial College, University of London, Research Report, 1989.

Paquete, L. F. and Fonseca, C. M., "A study of examination timetabling with multiobjective evolutionary algorithms," *Metaheuristics International Conference*, Porto, Portugal, 2001.

Pendharkar, P. C. and Rodger, J. A., "Nonlinear programming and genetic search application for production scheduling in coal mines," *Annals of Operations Research*, vol. 95, no. 1, pp. 251-267, 2000.

Pérez, E., Herrera, F. and Hernández, C. "Finding multiple solutions in job shop scheduling by niching genetic algorithms," *Journal of Intelligent Manufacturing*, vol. 14, no. 3, pp. 323-339, 2003.

Pinaki M. and Elizabeth M. R., *Genetic Algorithms for VLSI Design, Layout & Test Automation*, New Jersey: Prentice hall, 1999.

Podgorelec, V. and Kokol, P., "Genetic algorithm based system for patient scheduling in highly constrained situations," *Journal of Medical Systems*, vol. 21, no. 6, pp. 417-427, 1997.

Ponnambalam, S. G., Aravindan, P. and Chandrasekaran, S., "Constructive and improvement flow shop scheduling: An extensive evaluation," *Production Planning & Control*, vol. 12, no. 4, pp. 335-344, 2001a.

Ponnambalam, S. G., Aravindan, P. and Sreenivasa Rao, P., "Comparative evaluation of genetic algorithm for job shop scheduling," *Production Planning and Control*, vol. 12, no.6, pp. 560-574, 2001b.

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

Poon, P. W. and Carter, J. N., "Genetic algorithm crossover operators for ordering applications," *Journal of Computers and Operations Research*, vol. 22, no. 1, pp. 135-147, 1995.

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

Potvin, J. Y., Kervahut, T, Garcia and B, Rousseau, J. M., "The vehicle routing problem with time windows - part I: Tabu search," *INFORMS Journal on Computing,* vol. 8, no. 2, pp. 158-164, 1996.

Potvin, J. Y, Kervahut, T, Garcia B. and Rousseau, J. M., "A Tabu search for the vehicle routing problem with time window,", Centre de Recherche sur les Transports, University de Montreal, Canada, Technical Report CRT-855, 1993.

Pretolani, D., "A directed hypergraph model for random time dependent shortest paths," *European Journal of Operational Research*, vol. 123, no. 2, pp. 315-324, 2000.

Prinetoo, P. M. R. and Reorda, M. S., "Hybrid genetic algorithms for the traveling salesman problem", *Fifth International Conference on Genetic Algorithms*, pp. 559-566, 1993.

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

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

Rego, C., “Node ejection chains for the vehicle routing problem: sequential and parallel algorithms”, *Parallel Computing*, vol. 27, no. 3, pp. 201-222, 2001.

Rego, C., “A subpath ejection method for the vehicle routing problem,” *Management Science*, vol. 44, no. 10, pp. 1447-1459, 1998.

Rego, C. and Roucairol, C., “Using Tabu search for solving a dynamic multi-terminal truck dispatching problem,” *European Journal of Operational Research*, vol. 83, no. 2, pp. 411-429, 1995.

Reimann, M. and Doerner, K., “Savings based ants for large scale vehicle routing problems,” in *Operations Research Proceedings 2002*, Berlin/Heidelberg: Springer, pp. 205-210, 2002.

Renaud, J. and Boctor, F. F., “A sweep-based algorithm for the fleet size and mix vehicle routing problem,” *European Journal of Operational Research*, vol. 140, no. 3, pp. 618-628, 2002.

Ribeiro, C. C. and Hansen, P., *Essays and Surveys in Metaheuristics*, Boston: Kluwer, 2002.

Richardson, J.T., M.R. Palmer, G. Liepins and M. Hilliard, “Some guidelines for genetic algorithms with penalty functions,” In *Proceedings of 3rd International Conference on Genetic Algorithms*, ed. by J.D. Schaffer, Morgan Kaufmann, CA, pp. 191-197, 1989.

Rochat, Y. and Tailard, E. D., “Probabilistic diversification and intensification in local search for vehicle routing problem,” *Journal of Heuristic*, vol. 1, no. 1, pp. 147-167, 1995.

Roman Nossal, “An evolutionary approach to multiprocessor scheduling of dependent tasks,” *Future Generation Computer Systems*, vol. 14, no. 5-6, pp. 383-392, 1998.

Rossi, A. and Dini, G., “Dynamic scheduling of FMS using a real-time genetic algorithm,” *International Journal of Production Research*, vol. 38, no. 1, pp. 1 - 20, 2000.

Rousseau, L. M., Gendreau, M. and Pesant, G., “Using constraint-based operators to solve the vehicle routing with time windows,” *Journal of Heuristics,* vol. 8, no. 1, pp. 43-58, 2002.

Runarsson, T. P. and Jonsson, M. T., “Genetic production systems for intelligent problem solving,” *Journal of Intelligent Manufacturing*, vol. 10, no. 2, pp. 181-186, 1999.

Salhi, S. and Sari, M., “A multi-level composite heuristic for the multi-depot vehicle fleet mix problem,” *European Journal of Operational Research*, vol. 103, no. 1, pp. 95-112, 1997.

Savelsbergh, M. W. P., “Local Search for routing problems with time windows”, *Annals of Operations Research*, vol. 4, pp. 285-305, 1985.

Schaffer, J.D., “Multi-objective optimization with vector evaluated genetic algorithms,” In *Proceedings of the 1st International Conference on Genetic Algorithms: Genetic Algorithms and their Applications*, Lawrence Erlbaun, pp. 93-100. 1985.

Schonberger, J., Mattfeld, D. C. and Kopfer, H., "Automated timetable generation for rounds of a table-tennis league," in *Proceedings of the Congress on Evolutionary Computation 2000*, vol. 1, pp. 277-284, 2000.

Schulze J. and Fahle T, "A parallel algorithm for the vehicle routing problem with time window constraints," *Annals of Operations Research*, vol. 86, pp. 585-607, 1999.

Shackelford, M. and Corne, D., "Collaborative evolutionary multi-project resource scheduling," in *Proceedings of the Congress on Evolutionary Computation 2001*, vol. 2, pp. 1131-1138, 2001.

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

Shaw, P., "Using constraint programming and local search methods to solve vehicle routing problems," *Principles and Practice of Constraint Programming - CP98, Lecture Notes in Computer Science*, Maher, M. and Puget, J.-F. (eds.), New York: Springer-Verlag, pp. 417-431, 1998.

Shaw, K. J., Lee, P. L., Nott, H. P. and Thompson, M., "Genetic algorithms for multiobjective scheduling of combined batch/continuous process plants," in *Proceedings of the Congress on Evolutionary Computation 2000*, vol. 1, pp. 293-300, 2000.

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

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

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

Su, C- T and Shiue, Y. R., "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.

Swihart, M. R. and Papastavrou, J. D., "A stochastic and dynamic model for the single-vehicle pick-up and delivery problem," *European Journal of Operational Research*, vol. 114, no. 3, pp. 447-464, 1999.

Syswerda, G., "Schedule optimization using genetic algorithms," in Davis, L. (Ed.) *Hand book of Genetic Algorithms*, New York: Van Norstrand Reinhold, pp. 1332-1349, 1991.

Taillard, E., Badeau, P., Gendreau, M., Guertin, F. and Potvin, J. Y., "A Tabu search heuristic for the vehicle routing problem with soft time windows," *Transportation Science*, vol. 31, no.2, pp. 170-186, 1997.

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

Tan, K. C., Lee, L. H. and Ou, K., "Hybrid genetic algorithms in solving vehicle routing problems with time window constraints," *Asia-Pacific Journal of Operational Research*, vol. 18, no. 1, pp. 121-130, 2001b.

Tan, K. C., Lee, T. H. and Khor, E. F., "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.

Tan, K. C., Lee, L. H. and Ou, K., "Artificial intelligence techniques in solving vehicle routing problems with time window constraints," *Engineering Applications of Artificial Intelligence*, vol. 14, pp. 825-837, 2001d.

Tan, K. C., Lee, T. H. and Ou, K. and Lee, L. H., "A messy genetic algorithm for the vehicle routing problem with time window constraints," *IEEE Congress on Evolutionary Computation*, pp. 679-686, 2001e.

Tan, K. C., Khor, E. F., Cai, J., Heng, C. M. and Lee, T. H., "Automating the drug scheduling of cancer chemotherapy via evolutionary computation", *Artificial Intelligence in Medicine,* vol. 25, pp. 169-185, 2002.

Tang, L. and Liu, J. Y., "A modified genetic algorithm for the flow shop sequencing problem to minimize mean flow time," *Journal of Intelligent Manufacturing*, vol. 13, no. 1, pp. 61-67, 2002.

Tapan P. Bagchi, *Multiobjective Scheduling by Genetic Algorithms*, Kluwer Academic Publishers, Massachusetts, 1999.

Tavares J., Pereira F. B., Machado P. and Costa E., "On the influence of GVR in vehicle routing," *ACM Symposium on Applied Computing (SAC 2003)*, Florida, USA, March, 2003.

Thangiah, S. R., Osman, I. H., and Sun, T., "Hybrid genetic algorithm, simulated annealing and Tabu search methods for vehicle routing problems with time windows," Computer Science Department, Slippery Rock University, Technical Report SRU CpSc-TR-94-27, 1994.

Thangiah, S. R., “An adaptive clustering method using a geometric shape for vehicle routing problems with time windows,” *Proceeding of the Sixth International Conference on Genetics Algorithm*, vol. 1, pp. 452-459, 1995.

Toth, P. and Vigo, D., *The Vehicle Routing Problem*, Philadelphia: SIAM, 2002.

Toth, P. and Vigo, D., “A heuristic algorithm for the symmetric and asymmetric vehicle routing problems with backhauls,” *European Journal of Operational Research*, vol. 113, no. 3, pp. 528-543, 1999.

Tsujimura, Y. and Gen, M., “Parts loading scheduling in a flexible forging machine using an advanced genetic algorithm,” *Journal of Intelligent Manufacturing*, vol. 10, no.2, pp. 149-159, 1999.

Turkcan, A. and Akturk, M. S., “A problem space genetic algorithm in multiobjective optimization,” *Journal of Intelligent Manufacturing*, vol. 14, no. 3, pp. 363-378, 2003.

Tuzun, D. and Burke, L. I., “A two-phase tabu search approach to the location routing problem,” *European Journal of Operational Research*, vol. 116, no. 1, pp. 87-99, 1999.

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

Van Veldhuizen, D. and Lamont, G. B., “Multiobjective evolutionary algorithms: analyzing the state-of-art,” *Evolutionary Computation,* vol. 8, no. 2, pp.125-147, 2000.

Wang, K., Löhl, T., Stobbe, M. and Engell, S., “A genetic algorithm for online-scheduling of a multiproduct polymer batch plant,” *Computers & Chemical Engineering*, vol. 24, no. 2-7, pp. 393-400, 2000.

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

Wang, L. and Zheng, D. -Z., “A modified genetic algorithm for job shop scheduling,” *The International Journal Of Advanced Manufacturing Technology*, vol. 20, no. 1, pp 72 - 76, 2002.

Wang, W. and Brunn, P., “An effective genetic algorithm for job shop scheduling,” *International Journal of Mechanical Engineering*, vol. 214, part B, pp. 293-300, 2000.

Wen, C. and Eberhart, R., C., “Genetic algorithm for logistics scheduling problem,” in *Proceedings of the Congress on Evolutionary Computation 2002* (CEC2002), pp. 512-516, 2002.

Whitley, D., Starkweather, T. and Fuquay, D., “Scheduling problems and traveling salesmen: The genetic edge recombination operator”, *Third International Conference on Genetic Algorithms*, San Mateo, CA, pp. 133-140, 1989.

Wiangtong, T., Cheung, P. Y. K. and Luk, W., “Comparing three heuristic search methods for functional partitioning in hardware–software codesign,” *Design Automation for Embedded Systems*, vol. 6, no. 4, pp. 425-449, 2002.

Wilson, P.B. and M.D. Macleod, “Low implementation cost IIR digital filter design using genetic algorithms,” In *IEE/IEEE Workshop on Natural Algorithms in Signal Processing*, Chelmsford, U.K., 1993, pp. 4/1-4/8.

Wolpert, D. and Macready, G., "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67-82, 1996.

Wu, T. H., Low, C. and Bai, J. W., "Heuristic solutions to multi-depot location-routing problems," *Computers and Operations Research*, vol. 29, no. 10, pp. 1393-1415, 2002.

Xu, Y. L., Lim, M. H. and Er, M. J., "Investigation on genetic representations for vehicle routing problem", *IEEE International Conference on Systems, Man and Cybernetics*, vol. 4, pp. 3083-3088, 2005.

Yamada, T. and Nakano, R., "Scheduling by genetic local search with multi-step crossover," in *Fourth International Conference On Parallel Problem Solving from Nature* (PPSN IV), Berlin, Germany, pp. 960-969, 1996.

Yang, J. T., Huang, H-D. and Horng, J. T., "Devising a cost effective baseball scheduling by evolutionary algorithms," in *Proceedings of the Congress on Evolutionary Computation 2002* (CEC 02), vol. 2, pp. 1660-1665, 2002.

Yang, W. H., Mathur, K. and Ballou, R. H., "Stochastic vehicle routing problem with restocking," *Transportation Science*, vol. 34, no. 1, pp. 99-112, 2000.

Yellow, P., "A computational modification to the saving method of vehicle scheduling," *Operational Research Quart*., vol. 21, no. 2, pp. 281-283, 1970.

Yoon, S-H. and Ventura, J. A., "An application of genetic algorithms to lot-streaming flow shop scheduling," *IIE Transactions*, vol. 34, no. 9, pp. 779-787, 2002.

Youssef, H., Brigitte, C.-M. and Noureddine, Z., "Lower bounds and multiobjective evolutionary optimization for combined maintenance and production scheduling in job shop," in *Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation 2003* (ETFA '03), vol. 2, pp. 95-100, 2003.

Zbigniew, J. Czech and Piotr, C., "Parallel simulated annealing for the vehicle routing problem with time windows," Silesia University of Technology, Technical report, 2001.

Zhang, J. and Kwon, W. H., "Scheduling and optimization for a class of multi-stage hybrid manufacturing systems," in *Proceedings of the IEEE International Conference on Control Applications 2001* (CCA '01), pp. 487-491, 2001.

Zhang, L., Wang, L. and Tang, F., "Order-based genetic algorithm for flow shop scheduling," in *Proceedings of International Conference on Machine Learning and Cybernetics 2002*, vol. 1, pp. 139-144, 2002.

Zhao, C. and Wu, Z. "A genetic algorithm approach to the scheduling of FMSS with multiple routes," *International Journal of Flexible Manufacturing Systems*, vol. 13, no. 1, pp. 71-88, 2001.

Zhong, Q., Xie, T. and Chen, H., "Task matching and scheduling based on co-evolutionary model," in *Proceedings of the 3rd World Congress on Intelligent Control and Automation 2000*, vol. 3, pp. 1910-1915, 2000.

Zitzler, E., and Thiele, L., "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach", *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, 1999.

Zitzler, E., Evolutionary Algorithms For Multiobjective Optimization: Methods And Applications, PhD thesis, Swiss Federal Institute of Technology Zurich, 1999.

Appendix 1

Some of the routing solutions obtained by HMOEA in solving Solomon benchmark problems are given below.

$C_{1\text{-}01}$:

[90 87 86 83 82 84 85 88 89 91]
[13 17 18 19 15 16 14 12]
[81 78 76 71 70 73 77 79 80]
[67 65 63 62 74 72 61 64 68 66 69]
[5 3 7 8 10 11 9 6 4 2 1 75]
[20 24 25 27 29 30 28 26 23 22 21]
[32 33 31 35 37 38 39 36 34]
[43 42 41 40 44 46 45 48 51 50 52 49 47]
[57 55 54 53 56 58 60 59]
[98 96 95 94 92 93 97 100 99]

$C_{2\text{-}01}$:

[93 5 75 2 1 99 100 97 92 94 95 98 7 3 4 89 91 88 84 86 83 82 85 76 71 70 73 80 79 81 78 77 96 87 90]
[20 22 24 27 30 29 6 32 33 31 35 37 38 39 36 34 28 26 23 18 19 16 14 12 15 17 13 25 9 11 10 8 21]
[67 63 62 74 72 61 64 66 69 68 65 49 55 54 53 56 58 60 59 57 40 44 46 45 51 50 52 47 43 42 41 48]

$R_{1\text{-}04}$:

[72 75 56 23 67 39 55 4 25 54]
[53 58]
[88 62 11 63 64 49 19 7 52]
[89 60 83 17 45 8 46 36 47 48 82 18]
[27 69 76 3 79 29 24 68 80 12 26]
[50 81 78 34 35 71 65 66 30 70 1]
[95 92 37 98 93 59 99 84 5 96 94 13]
[97 42 14 44 38 86 16 61 85 91 100 6]
[2 57 15 43 87 41 22 74 73 21 40]

[31 10 90 32 20 9 51 33 77 28]

$R_{2\text{-}04}$:

[40 41 22 75 23 67 39 56 72 73 21 74 4 55 25 54 80 68 77 28]

[27 69 31 88 62 11 63 90 32 10 1 50 76 3 79 33 9 81 51 70 30 20 66 65 71 35 34 78 29 24 12 26]

[2 57 15 43 14 44 38 86 16 61 17 84 45 8 46 36 49 64 19 47 48 82 7 52 18 83 60 5 91 100 13 58]

[89 6 94 95 97 92 59 96 99 93 85 98 37 42 87 53]

$RC_{1\text{-}02}$:

[42 61 81 90]

[95 85 63 76 51 84 56 66]

[69 88 53 55 100 70]

[94 31 29 27 26 89 91 80]

[39 36 44 40 38 41 43 35 37 72]

[82 11 15 16 9 10 13 17 12]

[65 99 52 57 74 77 83]

[64 86 87 59 97 75 58]

[2 45 8 7 6 46 4 5 3 1]

[48 21 23 18 19 22 49 20 24 25]

[50 33 28 30 32 34 93 96]

[14 47 73 79 78 60 98]

[92 62 67 71 54 68]

$RC_{1\text{-}07}$:

[65 83 58 75 77 25 23 24]

[90 61 81 54 96]

[82 99 52 57 86 59 87 97 74]

[42 44 39 38 36 35 37 40 43 41]

[95 84 85 63 51 76 89 56 91]

[72 71 93 94 67 50 92 80]

[88 2 6 7 8 5 3 1 45 60 55]

[12 14 47 17 16 15 11 13 9 10]

[62 31 29 27 26 28 30 34 32 33]

[69 98 53 78 73 79 46 4 100 70 68]

[64 22 19 18 21 48 49 20 66]

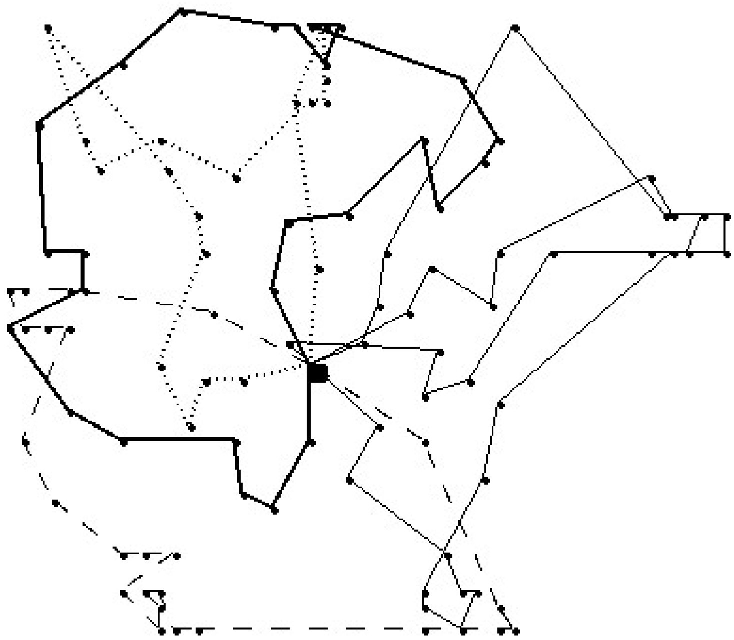
RC$_{2\text{-}07}$:

[92 95 67 62 33 30 28 29 31 71 72 42 44 40 38 39 41 61 81 90 94 96 93 50 34 27 26 32 89 56 91 80]

[82 11 15 16 47 14 12 73 79 7 6 2 8 5 45 46 4 3 1 43 36 35 37 54]

[69 98 88 53 99 52 86 75 59 87 74 57 22 20 49 48 24 66]

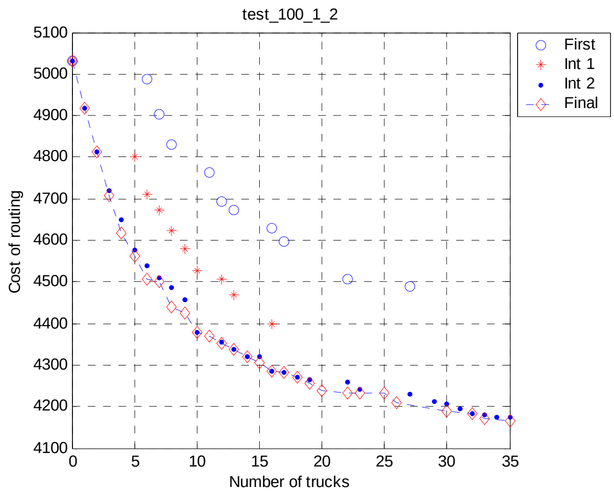
[65 83 64 51 84 85 63 76 21 18 19 23 25 77 58 97 13 9 10 17 78 60 55 100 70 68]

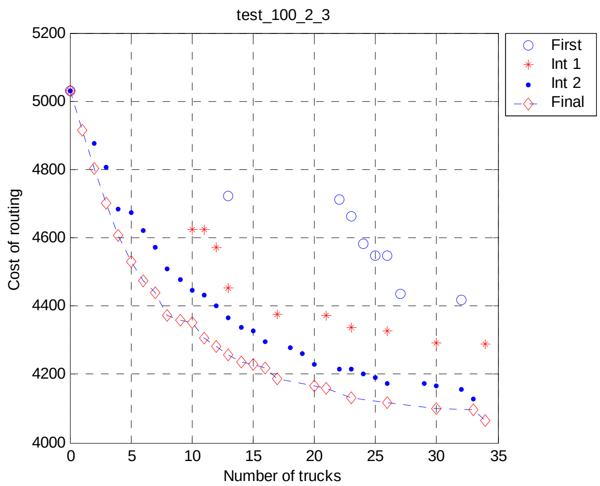


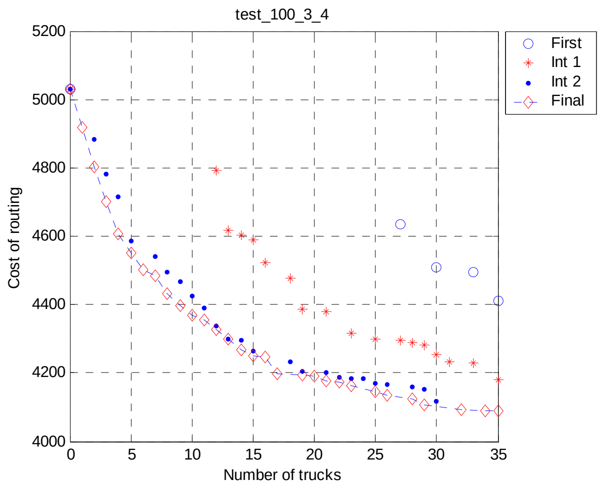
Solution for RC2-07: Black dots indicate 100 customer sites; the depot is represented by a black rectangle near the centre of map and routes are identified with different line styles.

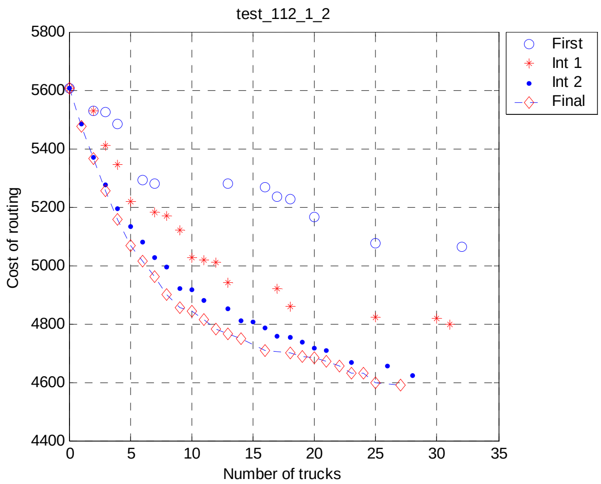
Appendix 2

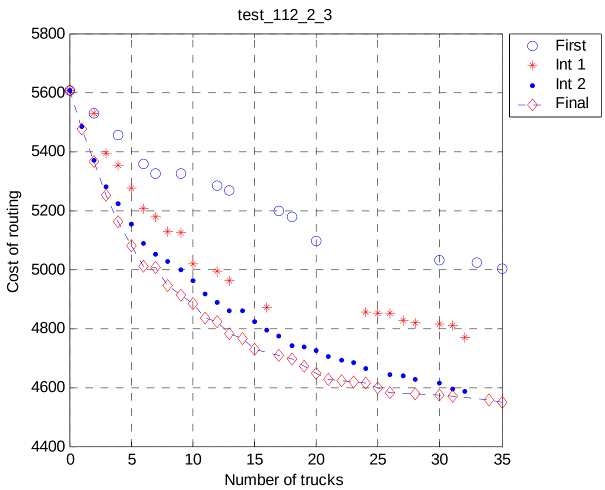
Enlarged views for the evolution progress of Pareto front for the 12 test cases in normal category. The initial generation (First), two intermediate generations (Int 1 and Int 2) and the final generation (Final) are plotted with different markers. As the evolution proceeds, the diversity of the population increases significantly and the non-dominated solutions gradually evolve towards the final trade-off curve. A dashed line connecting all the final non-dominated solutions is drawn for each test case, which clearly shows the final trade-off or routing plan obtained by the HMOEA.

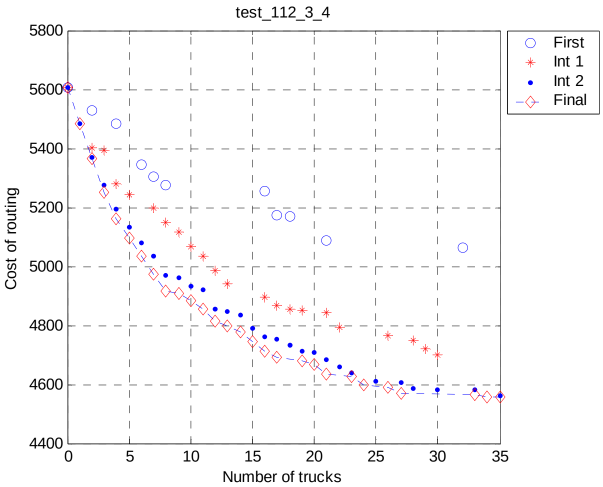


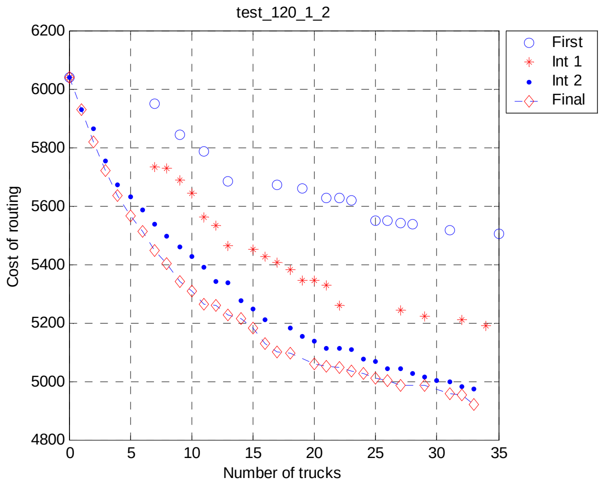


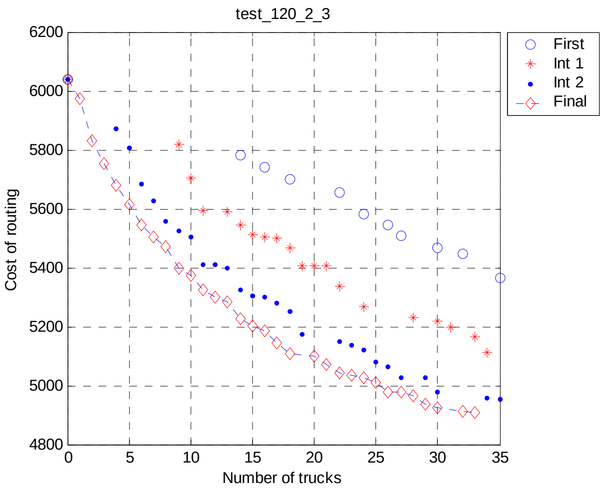

test_100_3_4
First
Int 1
Int 2
Final
Cost of routing
Number of trucks
5200
5000
4800
4600
4400
4200
4000
0
5
10
15
20
25
30
35

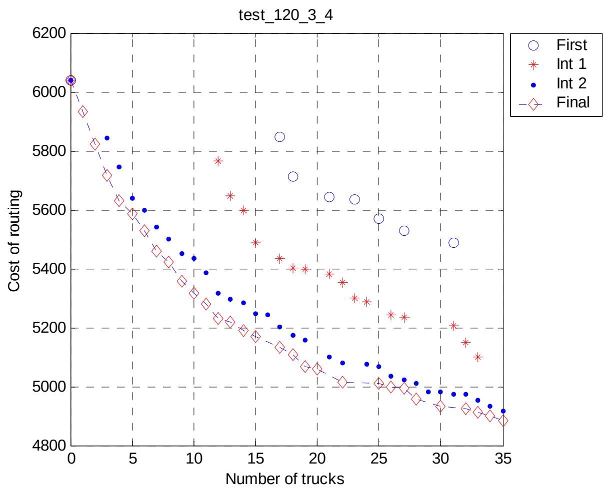

test_112_1_2
First
Int 1
Int 2
Final
Cost of routing
Number of trucks
5800
5600
5400
5200
5000
4800
4600
4400
0
5
10
15
20
25
30
35

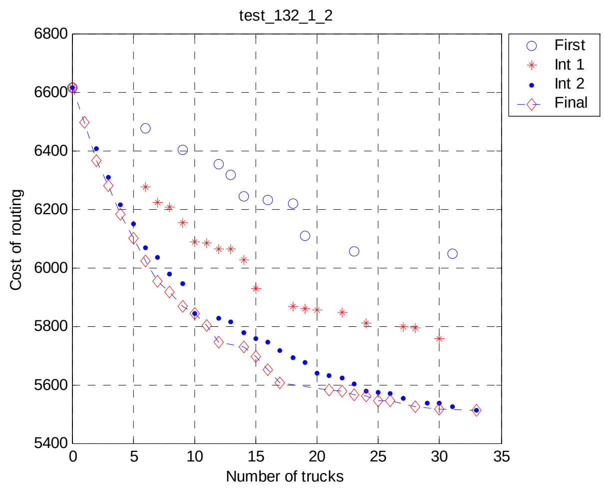

test_112_2_3
First
Int 1
Int 2
Final
Cost of routing
Number of trucks
5800
5600
5400
5200
5000
4800
4600
4400
0
5
10
15
20
25
30
35

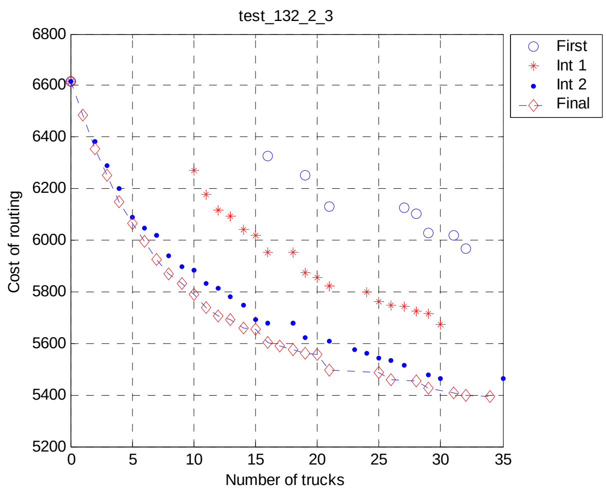

test_112_3_4
Cost of routing
Number of trucks
First
Int 1
Int 2
Final

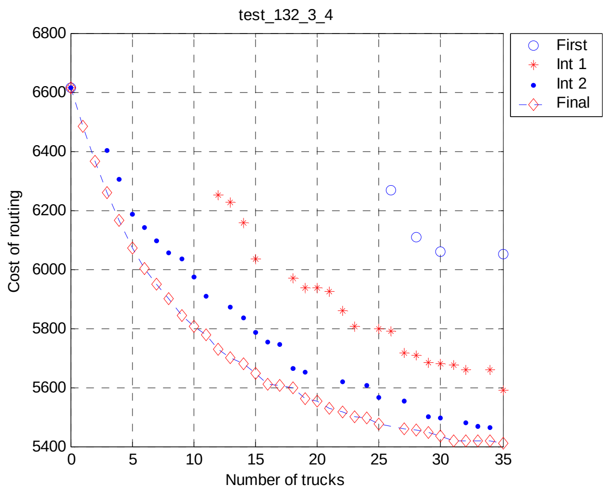

test_120_1_2
Cost of routing
Number of trucks
First
Int 1
Int 2
Final


test_120_2_3
Cost of routing
Number of trucks
First
Int 1
Int 2
Final


test_120_3_4
First
Int 1
Int 2
Final
Cost of routing
Number of trucks


test_132_1_2
First
Int 1
Int 2
Final
Cost of routing
Number of trucks


test_132_2_3
First
Int 1
Int 2
Final
Cost of routing
Number of trucks


test_132_3_4
First
Int 1
Int 2
Final
Cost of routing
Number of trucks
6800
6600
6400
6200
6000
5800
5600
5400
0
5
10
15
20
25
30
35

Author’s Publications

Journal publications

Tan, K. C., Chew, Y. H. and Lee, L. H., “A hybrid multiobjective evolutionary algorithm for solving vehicle routing problem with time windows”, *Computational Optimization and Applications*, vol. 34, no. 1, pp. 115-151, 2006.

Tan, K. C., Chew, Y. H. and Lee, L. H., “A hybrid multiobjective evolutionary algorithm for solving truck and trailer vehicle routing problems”, *European Journal of Operational Research*, vol. 172, no. 3, pp. 855-885, 2006.

Conference publications

Tan, K. C., Lee, T. H., Chew, Y. H. and Lee, L. H., “A hybrid multiobjective evolutionary algorithm for solving truck and trailer vehicle routing problems,” *IEEE Congress on Evolutionary Computation 2003*, Canberra, Australia, 8-12 December, pp. 2134-2141, 2003.

Tan, K. C., Lee, T. H., Chew, Y. H. and Lee, L. H., “A multiobjective evolutionary algorithm for solving vehicle routing problem with time windows,” *IEEE International Conference on Systems, Man and Cybernetics 2003*, Washington, D. C., USA, 5-8 October, pp. 361-366, 2003.

Tan, K. C., Prahlad, V., Lee, L. H. and Chew, Y. H., “A case study on vehicle routing problem with time windows,” *The First International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management* (HNICEM), Manila, Philippines, 2003.

Other publications

Lee, L. H., Tan, K. C., Ou, K. and Chew, Y. H., "Vehicle Capacity Planning System (VCPS): A case study on vehicle routing problem with time windows", *IEEE Transactions on Systems, Man and Cybernetics: Part A (Systems and Humans)*, vol. 33, no. 2, pp. 169-178, 2003.

Tan, K. C., Lee, T. H., Cai, J., and Chew, Y. H., "Automating the drug scheduling of cancer chemotherapy via evolutionary computation," *IEEE Congress on Evolutionary Computation 2002*, Honolulu, Hawaii, USA, pp. 908-913, 2002.