Multi-objective Genetic Algorithms for Multi-depot VRP with Time Windows

Sanjib Biswas

Submitted in partial fulfilment
of the requirements for the degree of

Master of Science

Department of Computer Science
Brock University
St. Catharines, Ontario

Abstract

Efficient routing and scheduling has significant economic implications for many real-world situations arising in transportation logistics, scheduling, and distribution systems, among others. This work considers both the single depot vehicle routing problem with time windows (VRPTW) and the multi-depot vehicle routing problem with time windows (MDVRPTW). An age-layered population structure genetic algorithm is proposed for both variants of the vehicle routing problem. To the best of the authors knowledge, this is first work to provide a multi-objective genetic algorithm approach for the MDVRPTW using well-known benchmark data with up to 288 customers.

Acknowledgements

I would like to thank the following people and organization for the support and believe in me:

- Prof. Beatrice M. Ombuki-Berman for her guidance, support and faith in me throughout the entire duration of my study and providing excellent supervision and funding.
- Prof. Brian Ross and Prof. Vladimir Wojcik for their shared knowledge and the participation in the supervisory committee.
- Cale Fairchild for his excellent technical support.
- Computer Science Department at Brock University for excellent education and environment to achieve this goal.
- To my loving parents and my two lovely brothers for their continuous support and believe.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The vehicle routing problem (VRP) is a widely studied combinatorial optimization problem. It was first introduced by Dantzig and Ramser [15] in 1959. A typical VRP can be defined as the problem of designing least-cost routes from a depot to a set of geographically dispersed customers with various demands. Each customer is to be serviced by only one vehicle and exactly once, and each vehicle has a limited capacity. The vehicle routing problem with time windows (VRPTW) is an extension of VRP; here time windows are associated with the customers, which means each customer provides a time frame within which the customer must be serviced. A vehicle may arrive early, but it must wait until the start of service time. If we consider more than one depot where the vehicles starts their journey then it is called multi-depot vehicle routing problem with time windows (MDVRPTW). In MDVRPTW, it is required that each vehicle must start and end at the same depot. In this research we studied a new variant of MDVRPTW, where a vehicle may not end at the same depot it started. Instead, it may end at the nearest depot to the last customer it serviced. We refer to this problem as the MDVRPTW under share depot or MDVRPTWSD.

The objective of the VRPTW, MDVRPTW, and MDVRPTWSD is to minimize the number of vehicles and the total travel distance to service all the customers without violating the capacity and time window constraints. There are other variants of the VRP and details of some of the popular variants can be found in [3] [10] [5]. The VRPTW, MDVRPTW, and MDVRPTWSD have received much attention due to the applicability of the time window constraints in real-world scheduling problems. Some of the well-known real-life applications of VRPTW and MDVRPTW are waste collection [48], fast-food routing [47] and many others.

The VRPTW, MDVRPTW, and MDVRPTWSD are classic examples of NP-complete multi-objective optimization problems. Obtaining the exact optimal so-

lution for this type of problems is computationally intractable. Thus we can rarely find an optimal solution within the reasonable time for a large instance. In fact, the smallest unsolved VRP instance (means that the best set of routes has not yet been found) has 50 customers and 8 vehicles (B-n50-k8 from [1]). No polynomial algorithms have been found for this type of problem. Various research has been applied to the problem using exact and approximation methods, but most of the research was done as a single objective problem. Details of some of the related works are given in Chapter 2.

Many real world problems are multi-objective by nature. In multi-objective optimization, problems have a number of objectives that are usually in conflict with each other, which means improving one objective may result in worsening others. Road transportation problems like VRPTW, MDVRPTW, and MDVRPTWSD are examples of a multi-objective optimization problem. This research studies these as a multi-objective optimization problem and implemented within the genetic algorithm (GA). Two of the objectives to be optimized are the number of vehicles, and the total distance traveled by the vehicles. Using the Pareto ranking technique, each of these objectives is kept independent. Though instance of VRPTW and MDVRPTW may have more than one non-dominated solution in the Pareto front, a user can choose which solution is more suitable according to their need. Some solutions may minimize the number of vehicles at the expense of traveled distance, and other reduces the distance at the number of vehicles. Minimizing the number of vehicles affects vehicle and labor cost while minimizing the distance affects the fuel cost and time.

GA is a population-based meta-heuristic that is inspired by Darwinian theory of the evolution "survival of the fittest." Previous analysis in [29] shows that GA is not performing better than other meta-heuristics. This could be because the solution gets stuck in local optima according to the research in [23]. To overcome this problem Hornby [23] proposed a multi-population evolutionary algorithm that resolves the issue of premature convergence. Some of the research on using Age-layered population structure (ALPS) [23, 2, 24, 25, 49] shows some advantages compare to other evolutionary algorithms (EAs).

1.1 Research Goals and Motivation

This thesis focused on VRPTW, MDVRPTW, and MDVRPTWSD. Our proposed ALPS based GA is compared with Ombuki *et al.* [37] and Keivan *et al.* [20] GA for VRPTW. Proposed ALPS based GA performed better than the existing research.

We also introduced MOPs for MDVRPTW and MDVRPTWSD as there exist no research on multi-objective optimization for MDVRPTW and MDVRPTWSD.

Although there has been a lot of research on other variants of the VRP [39, 31, 19, 6] [37] [38], but very little on VRPTW [37, 20] [23] and MDVRPTW [32, 36, 12, 13]. To the best of our knowledge, most of the research addressed the VRPTW and MDVRPTW problem as a single objective and tried to reduce the total distance covered by the vehicles. Reducing the number of vehicles to serve a set of customers can also be helpful for the industries. As reducing the number of vehicles means reducing the labor cost as well as vehicle cost. So, we are looking into the problem of VRPTW and MDVRPTW as a multi-objective optimization problem where two objectives are reducing total distance and reducing the number of vehicles. Literature [53, 16, 43] shows that EAs has a tendency for a premature convergence and sometimes leads to a non-optimal solution. In order to overcome this situation, we are using ALPS which helps prevent premature convergence [23] [24, 25].

1.2 Structure of Thesis

Chapter 2 provides the background required for this research. It includes the description of the variants of the VRP that has been used in this thesis, a brief description of the genetic algorithm, (GAs), fitness function and multi-objective optimization (MOO) approach used in this research and the age layered population structure (ALPS). Chapter 3 provides the details of the proposed ALPSGA. Chapter 4 describes the experimental results. Finally, we present our conclusions and future works in chapter 5.

Chapter 2

Background

2.1 Introduction

This chapter provides the relevant background on VRP, their variants, and formulation, and then a literature review of past works on the problem afterwards. Lastly, an overview of the genetic algorithm, age-layered population structure, and multi-objective optimization is given.

2.2 Vehicle Routing Problem

The vehicle routing problems (VRPs) are well-known combinatorial optimization problems that usually involve scheduling in constrained environments. A VRP can be defined as a problem of designing least cost routes from a central depot to service a set of customers with known demands. Each customer is to be serviced exactly once by only one vehicle, and each vehicle has a limited capacity. Figure 2.1 depicts a typical vehicle routing problem which can be formally [44] defined as:

- Let $G = (V, A)$ be a directed graph, where V is the vertex set, and A is the edge set.
- The vertex set V is divided into two subsets V_{cus} and V_{dep} where $V_{cus} = v_0, v_1, ..., v_N$ denotes the set of customers and $V_{dep} = v_{n+1}, ..., v_{n+d}$ denotes set of depots.
- $E = \{(v_i, v_j) | v_i, v_j \in V; i \neq j\}$ is the set of edges, where each edge represent the path from i to j.
- A cost matrix C of non-negative distances c_{ij} between customers v_i and v_j usually defined by the Euclidean distance from v_i and v_j.

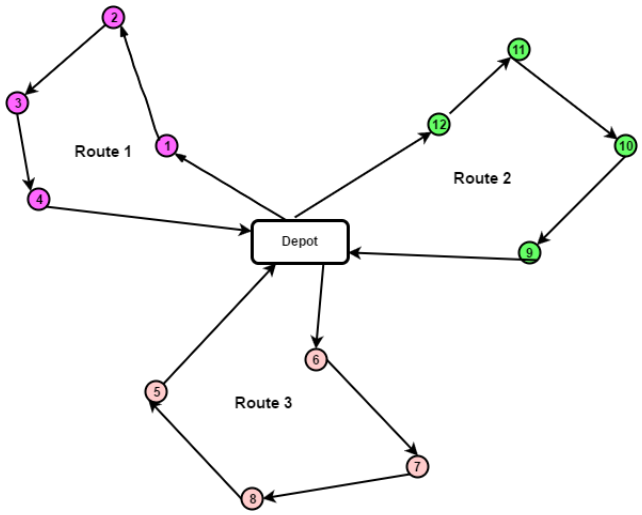

Figure 2.1: An example of a solution of the Vehicle Routing Problem with Time Windows with 1 depot, 12 customers and 3 vehicles

- A fleet of vehicles, where each vehicle has a certain capacity Q.

The goal of VRP is to find a set of routes with minimal cost, such a way that each customer is visited exactly once. This research are based on the following objectives and constraints.

Objectives

- Minimize the total number of vehicles/routes used to service all the customers.
- Minimize the total cost/distance travel by all the vehicles.

Constraints

- All routes must start and end at the depot.
- Each customer must be serviced exactly once by only one vehicle.
- In a route, the sum of demands of the customers cannot exceed the maximum capacity Q of a vehicle.

There are various well-known variants [3] [10] [5] of the VRP and in the next section, we will talk about two of the variants used in this research.

2.3 Vehicle Routing Problem With Time Windows

The vehicle routing problem with time windows (VRPTW) is one of the variants of the VRP, here a time window is associated with the customer. Each customer must be serviced within that time window. If a vehicle come before the start time, then it has to wait until the start of service time is possible, but a vehicle can not come after the end time. In the soft time window model, a vehicle is allowed to service a customer beyond the time window constraint with a penalty. However, in this work, and as commonly done, we focus on the hard time window model.

2.3.1 VRPTW Formulation

The VRPTW is represented [37] by directed graph $G = (C, A)$ and a set of homogeneous vehicles. Here C represent the set of customers. Each vehicles exit and return to the same depot, so node 0 and $n + 1$ represent the depot in the graph. The set of customers is represented by N. The set A represent all possible connection within the graph. Each route start at 0 and end at $n + 1$. Each edge $(i, j) \in A$ associated with a cost C_{ij} and time t_{ij} in a route. Each customer i, $i \in C$ has a service time and a demand d_i. Each customer also has a time window $[e_i, l_i]$, and all customers must be served within that time window, where e_i is the service start time, and l_i is the end time. A vehicle may come before the start time window (e_i), but cannot service the customers until the possible start time. However, no vehicle is allowed past the end time window, l_i. Vehicles must leave and return to the depot within the depot time constraint $[e_0, l_0]$. We assumed there is no waiting time in a depot and all the vehicles start at time 0.

The VRPTW model has mainly two types of decision variables, x and s. For each arc (i, j), $(i, j) \in A$, if $i \neq j$, $i \neq n+1$, $j \neq 0$, and for each vehicle k, the variable x_{ijk} will be 1 if vehicle k goes from vertex i to vertex j, otherwise 0. If vehicle k service customer i, $i \in C$, then variable S_ik denotes the time of the start of service by vehicle k, otherwise S_ik has no meaning. The main objective of the VRPTW model is to service all the customers C using the vehicles V such way that following objectives are achieved, and the constraints observed.

- Minimize the total number of vehicles/routes used to service all the customers.
- Minimize the total distance travel by all the vehicles.

Constraints

- Each customer must be serviced by only one vehicle and exactly once.
- In a route, the sum of demands of the customers cannot exceed the maximum capacity Q of a vehicle.
- Time window constraint should be observed.
- Each vehicle start at vertex 0 and end at the vertex n+1.

Mathematically [37] we can defined the VRPTW model in the following way:

$$min \sum_{k \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ijk}; \tag{1}$$

such that;

$$\sum_{k \in V} \sum_{j \in N} x_{ijk} = 1 \qquad \forall i \in C \tag{2}$$

$$\sum_{i \in C} d_i \sum_{j \in N} x_{ijk} \leqslant q \qquad \forall k \in V \tag{3}$$

$$\sum_{j \in N} x_{0jk} = 1 \qquad \forall k \in V \tag{4}$$

$$\sum_{i \in N} x_{ihk} - \sum_{j \in N} x_{hjk} = 0 \qquad \forall h \in C \qquad \forall k \in V \tag{5}$$

$$\sum_{i \in N} x_{i,n+1,k} = 1 \qquad \forall k \in V \tag{6}$$

$$s_{ik} + t_{ij} - K(1 - x_{ijk}) \leqslant s_{jk} \qquad \forall i \in N, \forall j \in N, \forall k \in V \tag{7}$$

$$0t_i \leqslant S_{ik} \leqslant ct_i \qquad \forall i \in N, \qquad \forall k \in V \tag{8}$$

$$x_{ijk} \in 0, 1 \qquad \forall i \in N, \qquad j \in N, \qquad \forall k \in V \tag{9}$$

Where $V = 1, 2, ...k$ vehicles
$C = 1, 2, ...n$ customers size
$0, n + 1$ is the depot
$N = 0, 1.....n, n + 1$ is the node size

d_i client i demand
a_i client i open time
b_i client i close time
q_k vehicle k capacity
t_{ij} client i from j time
s_{ij} client i take k service time

The objective function (1) gives that the total travel cost should be minimized. Function (2) states that each customer must be serviced by one vehicle and exactly once. Constraint set (3) ensures that vehicle capacity must be observed. The constraints set (4),(5),(6) states the each vehicle leaves from depot 0, visit customers and return to depot again. Equation (7) states that if vehicle K travel from i to j then, it can not arrive at j before $s_{ij} + t_{ij}$. Constrain (8) indicates the time window constraint must be observed and (9) gives the integrality constraints.

2.3.2 Previous Work on VRPTW

VRPTW is a classical example of NP-complete [33] multi-objective optimization problem. Obtaining the exact optimal solution of this type of problem is computationally intractable. There are no polynomial algorithms known, and it is believed that no such algorithm exist [33]. So many authors proposed different solution methods based on exact and heuristic approach. Kohl's work is one of the most efficient exact methods for VRPTW [30]; it succeeded in solving various VRPTW instances with 100 customers. However, no algorithms have been developed that able to solve all the 100 customers instances optimally. Exact methods are more efficient when customers have very narrow time windows since there are fewer combinations of customers possible in a route. As a result, researchers have investigated the heuristics and meta-heuristics methods for VRPTW problem.

Several heuristics have been applied to solve VRPTW and can be found in [52, 56]. Meta-heuristics like GAs can be found in [21] [54]. Other meta-heuristics such as simulated annealing [8], Tabu search [46, 9, 11] and ant-colony optimization can be found in the literature [17]. These approaches are very well suited for this type of problem rather than the traditional techniques. Genetic algorithm based approach [37] presented two new crossover operators and showed that the new crossover is well suited for the VRPTW problem. Thangaiah *et al.* [54] proposed a clustered first, route second method genetic algorithm with local search technique. Comparative study of

the genetic algorithm, simulated annealing, and Tabu search are given in [55]. The rich literature to solve VRPTW in exact and heuristics solution approach deals with a single point of view, which is the cost minimization (total travel distance). Not many papers are available that deals with the multiple objectives, especially the ones that solve the problem in polynomial time.

In the multi-objective approach, Gehring *et al.* [18] introduced a two-stage model which first tries to minimize the number of vehicles using an evolutionary algorithm and secondly attempts to minimize the total distance by Tabu search. A two-phased Tabu search proposed by Potvin at al. [42], firstly it tries to reduce the number of vehicles by moving customers out of the routes, and secondly, it tries intra and inter-customer exchange to reduce the total travel cost. Another similar research by Gambardella *et al.* [51] studied a type of multi-objective implementation of the VRPTW, where the first objective is to reduce the total number of vehicles and second minimized the total distance. It should be noted that all the above works on VRPTW are biased towards the number of vehicles. As most of these works use the weighted sum fitness measure, the vehicle and distance are evaluated as a single objective function. The main advantage of this approach is that single solution is obtained as a result. The Pareto optimality approach is taken as a multi-objective fitness scheme where both the objectives have the same priority.

The concept of Pareto optimality for solving the multi-objective VRPTW optimization proposed by Tan *et al.* [51] that incorporates various heuristics for local search in the evolutionary process. Genetic algorithm based models proposed by Ombuki *et al.* [37] introduced different and efficient operators to produce the better result. A similar study in this area by Keivan *et al.* [20] that modified the operators introduced by Ombuki at al. [37] to produce some competitive results.

This research studies a bi-objective VRPTW which are modeled by Genetic Algorithm similar to [20] [37], and then ALPS model is introduced to see if the results get better than the existing one. In this study, simultaneous minimization of the number of vehicles and total travel distance are considered as the two objectives. As with all the multi-objective optimization (MOP's) using Pareto optimality, one main advantage is that the system does not specify either the number of vehicles or the total travel distance to take priority. Each of the objectives in Pareto ranking procedure is kept separate, hence maintaining the independence of the function.

2.4 Multi-Depot Vehicle Routing Problem With Time Windows

The multi-depot vehicle routing problem with time windows (MDVRPTW) is a generalization of the VRPTW problem of Section (2.3), where instead of one depot, several depots are considered as shown in Figure 2.2. The MDVRPTW problem is considered a practical description of transportation planning. For example, such as those found in large cities where traffic congestion is high, commercial enterprises such as gas industry[4] have more than one depot from which their stations are serviced.

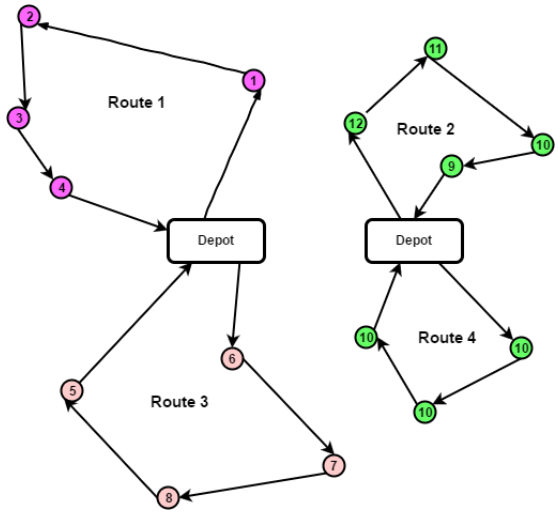

Figure 2.2: An example of a solution of the MDVRPTW with 2 depot, 15 customers and 4 vehicles

2.4.1 Problem Formulation MDVRPTW

The MDVRPTW is defined on a complete graph $G = (V, A)$, where

$$V = v_1, ..., v_N, v_{N+1}, ..., v_{N+M}$$

is the vertex set and

$$A = (v_i, v_j)|v_i, v_j \in V, i \neq j$$

is the arc set.

$$C = v_1, v_2, ..., v_N$$

represents the N customers and

$$D = v_{N+1}, v_{N+2}, ..., v_{N+M}$$

correspond to the M depots. Each vertex $v_i \in C$ has several non negative weights associated to it, namely, a demand q_i , a service time s_i, as well as a time window $[e_i, l_i]$, where, e_i is an earliest start time and l_i is a latest start time for the service. Further, in the depot vertex $v_i \in D$, there are no demand and service times, i.e. $q_i = s_i = 0$. Associated to each arc $(v_i, v_j) \in A$, there is a non negative travel cost c_{ij} , which represents the distance, travel time, or fee, etc. Finally, a fleet of $K = k_1, k_2, ..., k_L$ is the vehicles set and all the vehicles are homogeneous, where L is the number of vehicles. The objective of this problem remains same as the VRPTW section. Based on this graph, the MDVRPTW constraints must be satisfied as follows:

Constraints for MDVRPTW

- each vehicle starts from a depot and ends at the same depot;
- each customer is required to be served exactly once by exactly one vehicle;
- the service at each customer i must begin within the time window $[e_i, l_i]$. If a vehicle arrives customer i earlier than time e_i, it will wait.
- the total load and working duration (the sum of travel time, wait time and service time) of vehicle k does not exceed Q_k and T_k, respectively.

We are also looking at different set of constraints for MDVRPTW where instead of returning to the same depot a vehicle can go the nearest depot for the last customer it visited. We named this problem the multi-depot vehicle routing problem with time windows under share depot(MDVRPTWSD). Following are the set of constraints for the MDVRPTWSD problem.

Constraints for MDVRPTWSD

- each vehicle starts from a depot and ends at the depot closest to the last customer it visit.

2.4.2 Previous Work on MDVRPTW

The single depot vehicle routing problem with time windows (simply denoted as VRPTW) has been studied widely, and some of the literature is described in section 3.2. MDVRPTW which extends VRPTW, has attracted more attention recently because of the practical description of transportation planning. The focus of the MDVRPTW literature is based mainly on the heuristics and meta-heuristics. Cordeau *et al.* [11] applied the Tabu search to solve the problem of MDVRPTW in 2001, and later in 2004 introduce an improved version of the Tabu search [12]. In 2012 Cordeau and Maichberger [14] proposed a parallel iterated Tabu search to solve MDVRPTW problem. A variable neighborhood search for MDVRPTW was proposed by Polacek [41, 40]. Vidal *et al.* [57] designed a hybrid genetic algorithm and applied it to MDVRPTW. In 2012 Noori *et al.* [**?**] proposed a hybridization of genetic algorithm and Tabu search for MDVRPTW. In all the literature above, MDVRPTW is considered as a single objective problem, where the only objective is to reduce the total travel time. Although there is much literature to solve MDVRPTW as a single point of view, we could not find any research where MDVRPTW is considered as a multi-objective problem. Audrey [38] applied ALPS on capacitated multi-depot vehicle routing problem.

2.4.3 Previous Work on MDVRPTWSD

In MDVRPTW mentioned in Section 2.4, it is required that each vehicle must start and end at the same depot. A new variant of MDVRPTW, in which a vehicle may not end at the depot from where it starts studied in this section. When all the vehicles start and end at a different depot, then it is known as open vehicle routing problem [45]. Slight difference with MDVRPTWSD is that vehicles end at a depot and in open VRP vehicles end at a customer. Therefore, we can say MDVRPTWSD is considered as a combination of MDVRPTW and open VRP. In 2014 Jian *et al.* [28] first address the problem in which they proposed a hybrid genetic algorithm with an adaptive local search for MDVRPTWSDR. In [32] they introduced a new variant where the number of parking space at each depot is twice the vehicles available.

2.5 Genetic Algorithm

The genetic algorithm (GA) was introduced by Holland in 1975 [22]. This algorithm is a class of meta-heuristic based on the Darwinian concept of evolution “survival of the

fittest." In the GA a problem starts with a set of chromosomes called the population of individuals. Each chromosome in the population represents a potential solution. The initial population is generated either randomly or some heuristics. Solutions from one population are used to generate next set of the population with the hope that new population will be better than the original one. A selection mechanism is used to select individuals based on their fitness to form new individuals (offspring) via the genetic operations crossover and mutation. This process is repeated until a predefined condition (number of generations or the optimal solution) is met. Figure 2.3 and Algorithm 1 shows an outline of a simple GA.

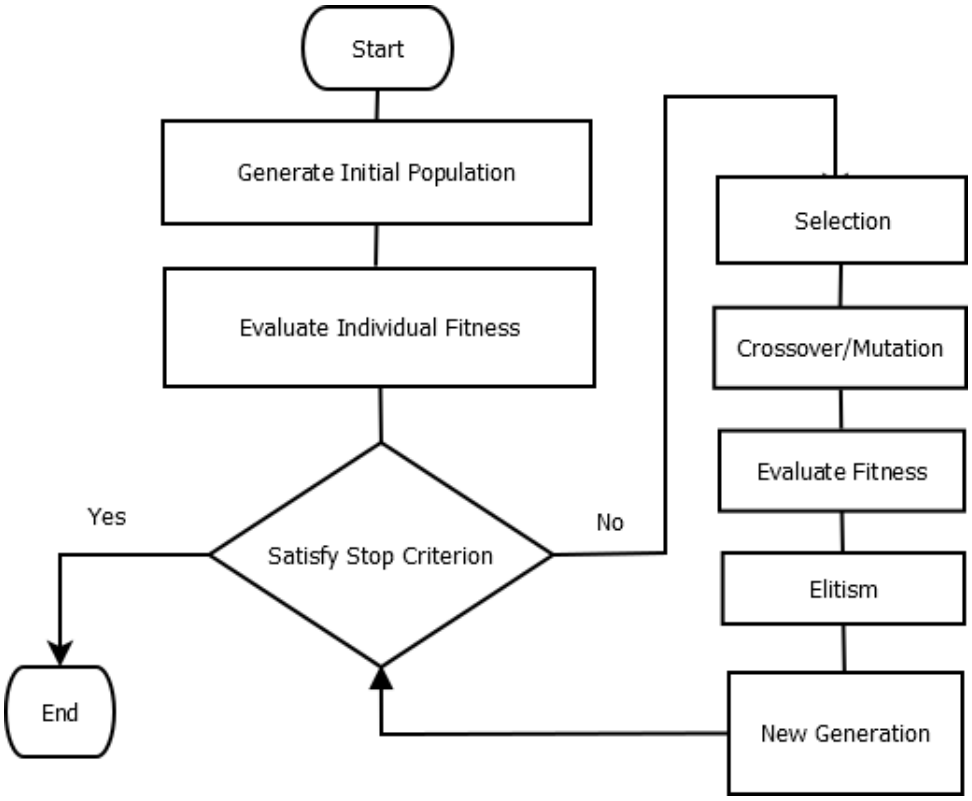

Figure 2.3: Genetic Algorithm (GA) work flow

Representation and initial population

For solving a problem using GA, it is necessary to represent each chromosome as a likely solution. The choice of this representation is one of the critical aspects of the GA. Various representation schemes such as integer, floating point numbers, binary strings, set-based representation, etc. are found in the literature [34, 7].

Selection

In this process, an individual is chosen based on the fitness from a population

Algorithm 1 Pseudocode for GA

```
1: procedure SimpleGA()
2:     Randomly generate initial population
3:     Get number of layers, age gap, AgeingScheme
4:     while !termination criterion do
5:         Compute the chromosome fitness
6:         Select elite population
7:         Select parents for reproduction
8:         Perform crossover operation to generate offspring
9:         Perform mutation to generate offspring
10:        newPopulation ⟸ individuals
11:        Population ⟸ newPopulation
12:    end while
13: end procedure
```

for the later breeding using the operations like crossover and mutation. Examples of selection process include tournament selection, roulette wheel, and rank selection.

Crossover

In the process of crossover, two selected individuals from the population are paired up for reproduction to create offspring by recombining their genes.

Mutation

Mutation helps a GA to break free from a fixed point in the search space. In mutation, only one chromosome is selected and the generated new offspring is then added to the new population.

Termination

In GA, termination criteria can be the pre-fixed number of generations or it can be a particular fitness level of a chromosome.

2.6 Age Layered Population Structure

The Age Layered Population Structure (ALPS) introduced by Hornby [23] as a technique for mitigating the problem of premature convergence in evolutionary algorithms. Hornby [24, 25] compared this approach with some other multi-population evolutionary algorithms like hierarchical fair competition (HFC) and adaptive-hierarchical fair competition (AHFC) and found that downside of this two algorithms resulted from the transfer of individuals form one layer to another based on fitness. Individuals with better fitness are highly observed. As a result, low fitness individuals are mostly not allowed to be transferred. In ALPS, an individual in the population uses a new

attribute called age to restrict the competition and reproduction between the individuals of the population. The main difference of ALPS and typical EAs is that in ALPS individuals are segregated into different layers (see Figure 2.3). With ALPS, several instances of the search are run independently, each in its own layer. The key properties of ALPS are:

- Multiple instances of the search are run independently, with each instance in its own age layer and having its own individuals.
- Each layer in the population has a maximum age limit, all the individuals' age in that layers must be lower than the maximum age.
- Each search in a given layer can select individuals from its own populations and the populations from the immediate next layer.
- At regular intervals, the first layers population is replaced with new random populations.

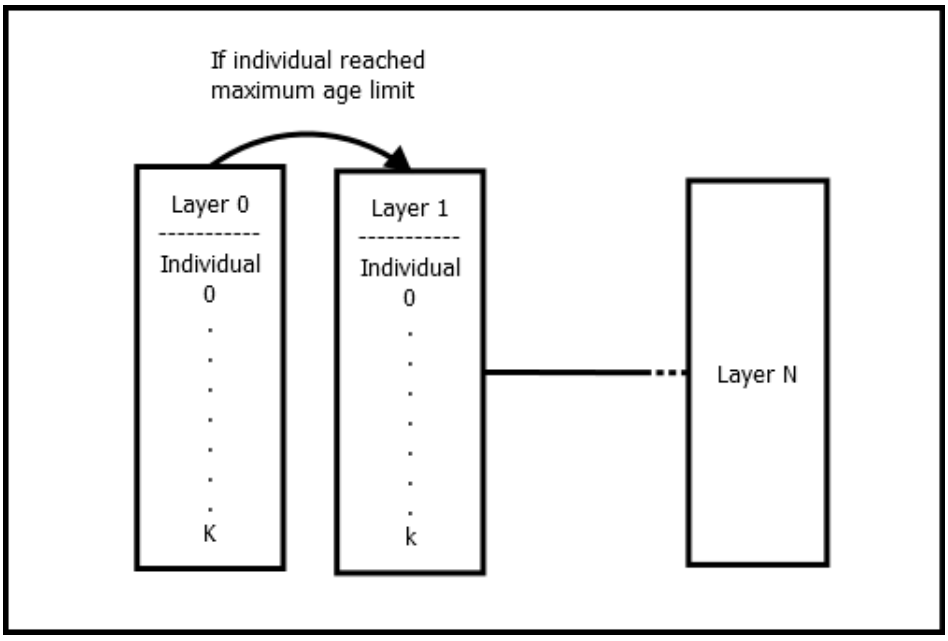

Figure 2.4: ALPS paradigm with N layers and k individuals in each layer

2.6.1 ALPS Algorithm

Similarity to GAs, ALPS also start with randomly generating the initial layer (layer 0). All other layers are then filled as the evolution progress. If the bottom layer reaches its maximum age, old populations are moved to the next layer, and the layers are filled with random populations again. This process continues until the stop criteria satisfied. Algorithm 2 shows a simple ALPS outline.

Algorithm 2 Pseudocode for ALPS

```
1: procedure ALPS()
2:     Read parameter file
3:     Get number of layers, age gap, AgeingScheme
4:     setLayers ⟸ InitialiseLayers(number of layers, age gap, ageingScheme)
5:     while !termination criterion do
6:         if (generation == 0||generation%layer0.agelimit == 0) then
7:             M ⟸ RandomlyCreateNewIndividuals()
8:             Else
9:             evolveIndividuals()
10:        end if
11:        if (generation>layer0.agelimit) then
12:            evolveIndividuals()
13:        end if
14:    end while
15: end procedure
```

```
1: procedure EvolveIndividuals()
2:     Perform elitism
3:     procedure PerformSelection()
4:         if (layer0) then
5:             Parents ⟸ selectparents(layer0)
6:             Else
7:             Parents ⟸ selectParents(currentLayer(C), (C-1)Layer, selection pres-
   sure)
8:         end if
9:     end procedure
10:    Perform Crossover
11:    Perform mutation
12:    if (individual.age==layer.ageLimit) then
13:        nextLayer ⟸ moveIndividual()
14:    end if
15: end procedure
```

2.6.2 Aging schemes

Different types of aging schemes [23] are available to set the age limit of a layer(see Table 2.1). In order to keep the number of layers manageable, each value in a layer is multiplied by an age gap parameter. For example, if we take the polynomial aging scheme [23] and the age gap value is 10, the maximum ages for the layers will be 10, 20, 40, 90, 160 and so on. Individuals within a layer are not allowed to evolve the maximum age for the layer. If an individual reaches the maximum allowed age, an attempt is made to move the individual to the next layer except for the last layer. According to Hornby [23], an individual in the last layer is allowed to stay there if it is a global optimum otherwise, it will be replaced by other individuals with higher fitness value from the lower layers. Introducing new individuals in the layer 0, allows the algorithm to search new parts of the fitness landscape. The evidence in [23] [24, 25] shows that ALPS helps to overcome the problem of premature convergence in the system.

Table 2.1: Ageing Schemes for ALPS [23]

Aging-Scheme	0	1	2	3	4
Linear	1	2	3	4	5
Fibonacci	1	2	3	5	8
Polynomial(n^2)	1	2	4	9	16
Exponential(2^n)	1	2	4	8	16

2.6.3 Previous Work on ALPS

ALPS is a diversity enhancing algorithm which allows new individuals in the system at regular interval and divides the total population into several layers. Other layers EA system such as hierarchical fair competition [26], adaptive hierarchical fair competition [27] usually group individuals based on the fitness of the individuals. ALPS groups the individuals based on their ages. The ALPS was introduced with genetic programming context [23], but Hornby compared it with a GA and proposed that ALPS can be used with any evolutionary algorithms. ALPS was compared on image operator problem [50] with Cartesian genetic programming and declared that ALPS perform better than typical CGP. Awuley conducted a comparative study on GP, ALPS, and feature selection ALPS (FSALPS) [2]. In [38], ALPSGA was declared better than the traditional GA when compared on capacitated MDVRP problem.

2.7 Multi-Objective Problems

Multi-objective optimization involves finding an optimal solution for multiple objective functions defined under a set of constraints. Example problems are design analysis, transportation trade-off, or any other system where one needs to find an optimal solution with trade-offs between two or more objectives. In most cases, these two or more objectives are conflicting to each other. In VRP, two objectives are the number of vehicles and the total cost. They define two independent objectives in a multi-objective problem hence in determining the fitness of an individual. The well-known fitness evaluation techniques are used in this research.

2.7.1 Weighted Sum

The weighted sum technique assigns a single fitness score to a multi-objective problem. With this technique, one of the objectives gets higher priority than others, as a result, produce a biased solution in general. As an example multi-objective vector $F(x)$ can be converted into a scalar problem by constructing a weighted sum of all the objectives. Fitness F(x) of a two objective (f1 and f2) problem is:

$$F(x) = W1.f1(x) + W2.f2(x)$$

where, W1 and W2 are the weights assigned to the individual objectives.

2.7.2 Pareto Ranking

Pareto efficiency is a state of multiple objectives in which it is impossible to say one individual is better than other without making one individual worse than other. It was first introduced by Vilfredo Pareto [37]. Unlike the weighted sum here each objective gets the same priority hence maintain the independence of each objective. When we get more than one solution in the Pareto front, it is user choice which objectives to consider most.

The Pareto scheme is incorporated in the GA by replacing the individual fitness in Pareto fitness. Individuals in the population are first ranked by theirs raw fitness score, and then these scores are replaced with the corresponding Pareto ranks. For example, given a vector of objectives $\vec{f} = (f_1, ..., f_k)$ subject to appropriate problem constraints, then vector $\vec{u}$ **dominates** $\vec{v}$ iff

$$\forall i \in (1, ..., k) : u_i \leq v_i \wedge \exists i \in (1, ..., k) : u_i < v_i, \textit{denoted by } \vec{u} \preceq \vec{v}$$

Algorithm 3 Pseudocode for Pareto Ranking [37]

```
1: procedure ParetoRank()
2:     Input Population
3:     procedure PerformParetoRanking()
4:         currentRank ⟸ 1
5:         Popsize ⟸ populationSize
6:         N ⟸ Popsize
7:         while (N ⩾ 1) do
8:             for (i=1 to N) do
9:                 if v_i is non-dominated then
10:                    rank(v_i)= currentRank
11:                end if
12:            end for
13:            for (j=1 to N) do
14:                if rank(v_j)= currentRank then
15:                    remove (v_j) from population
16:                    Popsize=Popsize-1
17:                end if
18:            end for
19:            currentRank=currentRank+1
20:            N=Popsize
21:        end while
22:    end procedure
23: end procedure
```

From the above definition, we can say a vector is dominated if and only if another vector is better in at least one objective and all other objectives are as good as the remaining objectives. As VRP is a minimization problem, ranks with lower values are the best solutions. Note that Pareto ranks are calculated with relative measurements, and therefore there is no single "best solution." Algorithm 3 gives Pseudocode for the Pareto ranking procedure.

Figure 2.5 gives a typical output of the Pareto ranking system. The process starts with assigning rank 1 to the non-dominated individuals in the population. These chromosomes are then removed, and the remaining non-dominated in the populations are assigned rank 2. This process is repeated until all the individuals in the population are ranked.

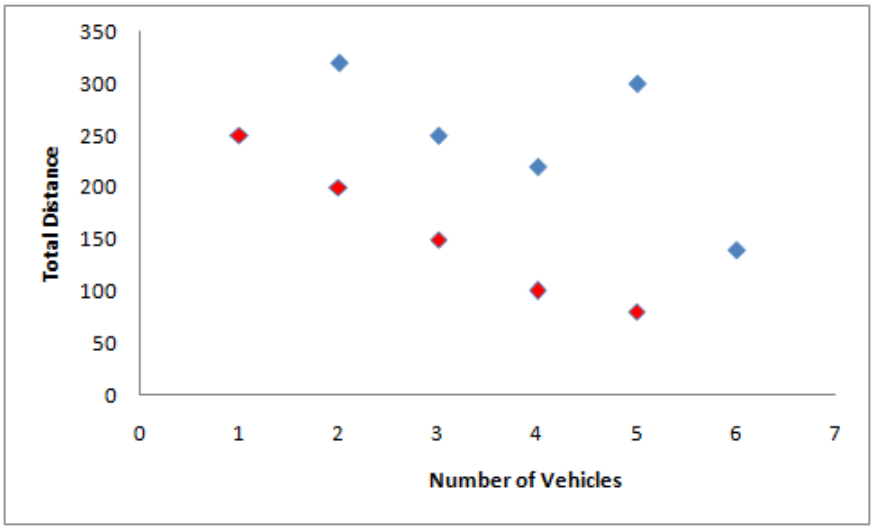

Figure 2.5: Typical output for Pareto ranking scheme where red squares are the rank 1 solutions

Chapter 3

System Design

3.1 Introduction

This chapter provides the implementation details of the proposed ALPSGA for the VRPTW problem. We also introduced a multi-objective optimization problem for the MDVRPTW problem. ALPS is incorporated to study the performance of GA compared with ALPSGA for MDVRPTW problem. A new variant of the MDVRPTW problem known as MDVRPTWSD is studied. Two new mutation techniques are developed to improve the performance of MDVRPTW and MDVRPTWSD problem. Background research and study of these problems is described briefly in Chapter 2.

3.2 ALPS-based Genetic Search for VRPTW

This section details the ALPS based GA system for VRPTW, fitness function, Pareto ranks and other GA parameters used. The basic difference in GA and ALPS based GA is that in ALPS we segregate the total number of populations in several layers and set the age limit for each layer. We then activate the GA in the lower level and carry out the process in every other layer. After subsequent evolutions, when the individuals reach their maximum age limit these individuals are then transferred to the next layer with the introduction of new randomly generated individuals in the lower level. Tournament selection and elitism are used throughout the GA for further evolutionary reproduction. Fig 3.1 outlines the ALPS based GA system.

3.2.1 ALPS Based GA

ALPS starts with the population split in a sequence of layers, with an upper limit of the maximum age of an individual that each layer can contain. Evolution of ALPS proceeds like typical GA, that is, the evolution of populations using crossover and selection operations on the chromosomes. However, unlike GA, there are two restrictions. First, the bottom layer of ALPS is replaced with randomly generated individuals at a regular interval (usually generation span). Secondly, reproduction of individuals is restricted to their own layer and from the layer immediately below them.

According to the first restriction stated above, individuals in layer 0 are replaced with randomly generated individuals. An aging scheme (see Table 2.1) is used to separate the individuals into several layers. These values are then multiplied by the age gap parameter to determine the maximum age limit per layer. Thus, with an age gap value of 50, new individuals are created in the first layer at generations 50, 100, 150, 200. According to the second restriction, reproduction of individuals for layer 0, parents are selected only from layer 0; for layer 1, parents are selected from individuals in layer 0 and 1; all other layers follow the same rule.

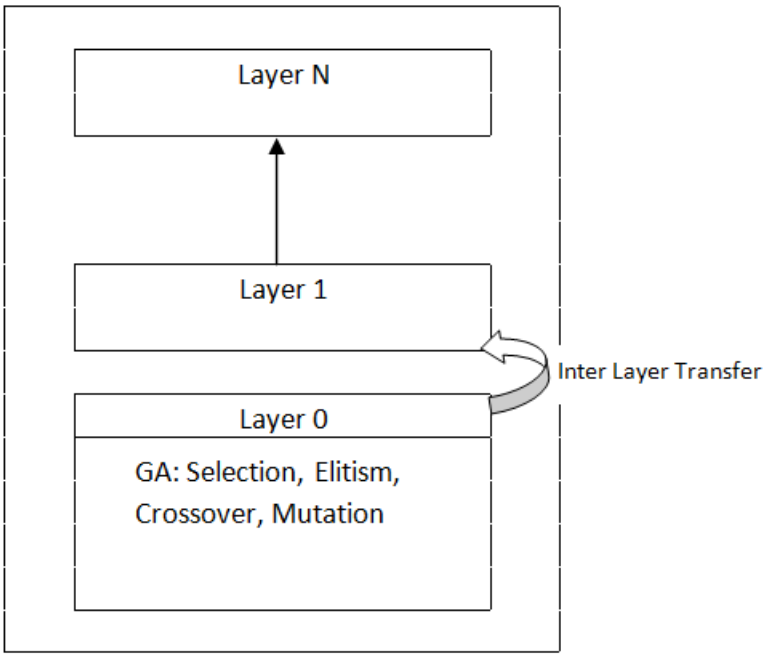

Figure 3.1: ALPS based GA for VRPTW

3.2.2 Chromosome Representation and Initial Population

In order to apply GA to a problem, we first need to determine the chromosome representation for the solution space. An indirect representation depicting a network configuration for a given VRPTW problem instance is given as an integer string of length N, where N represents the number of customers in that problem with. The sequence of chromosome genes representing the order of vehicle visitation at each customer stop in the route. The initial population of the ALPSGA is created by a random permutation of N customers. An example of a chromosome is given in Figure 3.2. This figure illustrates a VRPTW chromosome with 9 customers.

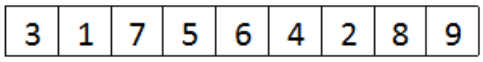

Figure 3.2: Chromosome representation of VRPTW with no delimiter showing start and end of route

3.2.3 Fitness Evaluation

Two different fitness evaluation schemes are used.

Weighted Sum Method

Weighted sum fitness function converts a multiple objective function into single objective by assigning weights to the individual objectives. In ALPSGA for VRPTW, the weighted fitness function $F(x)$ is,

$$F(x) = \alpha.|V| + \beta.\sum_{k\varepsilon V} D_k$$

Where α and β are the weights for the number of vehicles and total distance traveled by vehicles respectively. D_k describes the summation of total distance by all the routes in an individual. The weight value used for ALPSGA on VRPTW were established empirically as $\alpha = 100$ and $\beta = 0.001$.

Pareto Fitness Evaluation

As mentioned in Section 2.7.2, a Pareto ranking scheme works by replacing the chromosome raw fitness with Pareto ranks. These ranks are integer values that represent

the layers of dominant individuals in the population. The Pareto ranks are then used by the ALPSGA to generate the next population.

3.2.4 Elitism

Elitism is carried out in each generation at each layer in ALPSGA. It is incorporated so that the best individual is carried out into the next generation. This means that the best solution produced in a generation must be forwarded to the next generation. The elite population to transfer to the next generation is set to 4 in each layer.

3.2.5 Selection

A fitness based selection scheme called tournament selection is used to select parents for reproduction. In tournament selection, a set of K individuals are randomly selected from the current population, and the fittest individual is selected for reproduction. In this work, k is set to 4.

3.2.6 Crossover

In [37] we employed a problem specific crossover called best cost-route crossover (BCRC) which aimed at minimizing the number of vehicles, as well as total distance, traveled while checking the feasibility constraints. In that operation, two routes are selected randomly for the operation. Later in 2010 Keivan *et al.* [20] proposed best cost-best route crossover(BCBRC) which is very similar to BCRC[37] with minor differences. In BCBRC [20], a route is chosen according to the criteria of averaged cost over nodes instead of randomly in BCRC[37].

Figure 3.3 illustrates the creation of two offspring using BCRC. Let us consider P1 and P2 are two parents of customer size 9. RP1 and RP2 give corresponding sets of routes associated with P1 and P2, respectively. Firstly, from each parent, a route is chosen randomly. In this case, for P1, route with customers 5 and 6 is chosen, while for P2, route with customers 7 and 3 is picked. We then removed the customers from opposite parent from the given parent. In this case, for P1, customers 7 and 3 removed. The next step is to find the best possible locations for the removed customers in the children. In this case customers 3 and 7 needs to re-insert in child C1. Order of insertion of the removed customers is done arbitrarily. In this case customer 3 was first inserted in C1.

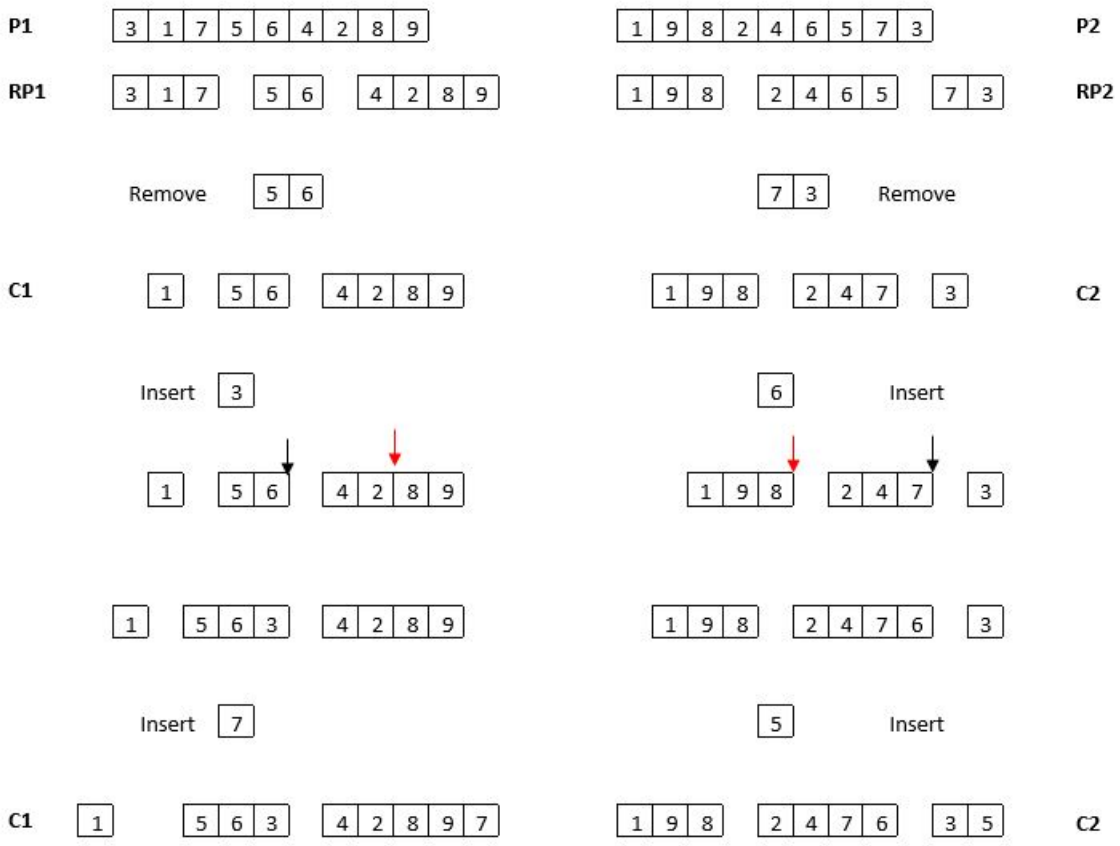

Figure 3.3: Best Cost Route Crossover for VRPTW

3.2.7 Mutation

We used a reassigning mutation which is similar to the intra-depot mutation described in [37]. Here, a customer is selected randomly from the chromosome and reinserted into a random position within the same chromosome. For example, if a route in a chromosome has 1,2,3,4,5 as customers, if customer 4 is selected, it could be reinserted into the route as 1,2,4,3,5.

3.2.8 Replacement strategy

The ALPSGA used the generational replacement strategy[23] within the layer. When an individual reaches its maximum age allowed according to the algorithm 2, we transfer it to the next level except for the last layer. In [2], [38] three different types of replacement have been discussed.In [38] Audrey conclude that best individual replacement is better for MDVRP problem. We have used the best individual replacement strategy to transfer or migrate population from one layer to another.

Firstly, the aged individuals from the lower level are added to the next upper level. The combined individuals are then sorted according to their fitness. We keep the same size of the population from the sorted list and rest of them are discarded. This technique ensures that best individuals are always kept.

3.3 Multi-objective Genetic Search for MDVRPTW

This section provides the details implementation of GA for MDVRPTW. We studied the problem as a multi-objective optimization problem. All other related works on MDVRPTW is described in Section 2.4.

3.3.1 GA for MDVRPTW

GA for multi-objective MDVRPTW starts with clustering of customers at nearest depot. Assigned customers are then used to create the initial random population. Evolution is carried out on the initial population to create next set of population involving tournament selection, crossover, and mutation. Evolution stops if a termination criterion is reached. The basic algorithm of GA is described in section 2.5, and proposed system is given in Figure 3.1.

3.3.2 Initial Depot Clustering

Initially, each customer is assigned to the nearest depot based on the Euclidean distance. As there is no capacity limit on each depot so we can assign as many customers as we want. During the assignment, customers are identified as the borderline customers if that customer is within a certain distance from more than one depot. In the mutation, we used these borderline customers similar to the work in [37].

3.3.3 Chromosome Representation and Initial Population

A chromosome in the MDVRPTW problem must specify the number of vehicles and its corresponding depot. We used an indirect representation of the chromosome that tells us the information above and also gives us the order of visitation. The MDVRPTW chromosome representation consists of n integer vectors, where n is the number of depots in that particular instance data. Each integer vector represents a depot and similar to the chromosome described in Section 3.2.2. Chromosome representation of MDVRPTW is shown in Figure 3.4.

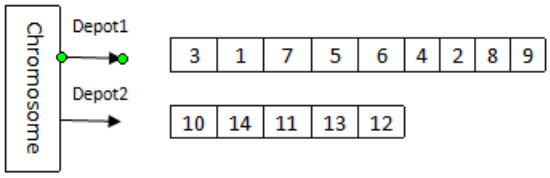

Figure 3.4: Chromosome representation of MDVRPTW with two depots with delimiter showing start and end of route

3.3.4 Fitness Evaluation

We used the same multi-objective techniques described in Section 3.2.3.

3.3.5 Elitism

Same as Section 3.2.4.

3.3.6 Selection

We have used the same selection model used in Section 3.2.5.

3.3.7 Crossover

The best cost route crossover (BCRC). It was first introduced in [37] and the same crossover we are using for VRPTW described in Section 3.2.6. For the MDVRP problem, this crossover was slightly changed in [20]. Figure 3.5 describe the implementation details of BCRC crossover for MDVRPTW problem. The following notation and algorithm describe the BCRC crossover. Given a population $P = p_1, p_2...p_n$ of chromosomes, where each chromosome $p_i = d_1, d_2...d_m$ consist of depots. Each depot $d_i = r_1, r_2, ...r_k$ consist of k routes. Each route k consist of not empty set of customers $r_i = c_1, c_2, ...c_l$. Following steps are taken to perform BCRC for MDVRPTW.

- Randomly select two parents p_1, p_2 from the population P.
- Randomly select a depot for both parent and select a random route from each parent under that depot. For example, route with customer 1,8 from p_1 and 6,7 from p_2.
- Remove customers 1,8 from p_2 and customers 6,7 from p_1.

- Reinsert the removed customers from each parent to form the offspring.
  - Select a removed customer from p_1 (like 1 from p_1) and insert it at each index in unremoved route of p_2 of selected depot.
  - Compute the cost of insertion at each index and the feasibility of insertion at each index. Create an ordered list of insertion with feasibility. If it breaks any of the constraints, that infeasibility is set to true.
  - If there are no index for a possible insertion in the unremoved route, then a new route is added.
  - Repeat the same procedure for all the removed customers at the selected depot.

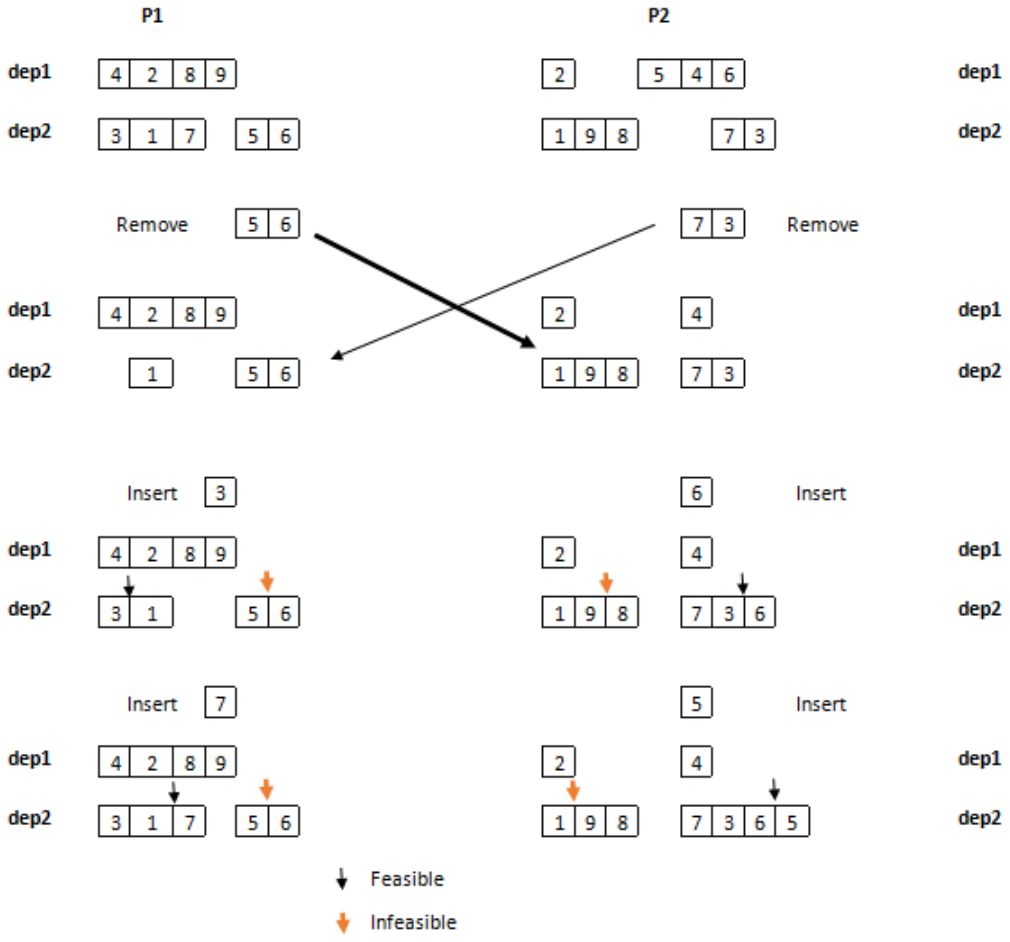

Figure 3.5: Best Cost Route Crossover for MDVRPTW

3.3.8 Mutation

We have used four mutations in this research. Re-assigning mutation like intra-depot and inter-depot mutation was inspired by single customer rerouting in [37]. We have proposed two new mutations technique for the MDVRPTW problem. Both of the new mutation does a local search to find the best possible location for the customer to insert.

Intra-depot Mutation

First, a random customer is removed from a random depot. It is then reinserted at a random position within that depot. For example, depot 1 of a chromosome has 2,3,4,5,6 as customers. If customer 4 is selected to remove from that depot, it could be re-inserted to that depot as 2,3,5,6,4.

Intra-depot Neighborhood Search

This neighborhood search is very efficient with complex problem constraints like MDVRPTW. A random customer is removed from the solution, but reinsertion is done by a particular rule to obtain a better solution. We followed a greedy insertion mechanism, where it does a local search within the depot to find the best possible location for that customers.

Inter-depot Mutation

As discussed in Section 3.3.2, during the process of initial depot clustering we identified some of the customers as borderline customers. Each of the borderline customers will have more than one depot associated with it; we called this list swappable customers. Although each customer is initially assigned to the nearest depot during the initialization, if any other depot is within a certain distance from a customer, that customer will be added to the swappable customers list. The list of such customers is made up with the equation 3.1.

$$[(dist(cust_i, d_i) - min)/min] <= BOUND \tag{3.1}$$

Here, $dist(cust_i, d_i)$ is the distance of c to the depot d_i, min is the minimum distance of c from the nearest depot and BOUND is a constant value, which is set according to [19].

Inter-depot Neighborhood Search

The inter-depot neighborhood mutation is similar to the intra-depot mutation described above. The main difference is that instead of choosing the same depot for reinserting we chose the best depot where the objective function minimizes the most.

3.4 Multi-objective ALPS based Genetic Search for MDVRPTW

This section provides the ALPS based GA system for MDVRPTW. In Section 3.2 we discuss how ALPS incorporated for VRPTW and in Section 3.3 we discussed GA system to solve MDVRPTW. ALPS based GA for MDVRPTW is similar to Section 3.3 except the GA within the layer. GA system designed in Section 3.3 is incorporated. All the fitness function, multi-objective optimization technique, GA parameters are similar to Section 3.3.

3.4.1 Chromosome Representation and Initial Population

We used the same chromosome we used in Section 3.3.3 and followed the same initialization as well.

3.4.2 Multi-objective Optimization

We used same multi-objectives techniques used in Section 3.2.3

3.4.3 Genetic Operations

We used BCRC crossover, intra and inter-depot mutation described in Section 3.3.7 and 3.3.8 respectively.

3.5 Multi-objective ALPS based Genetic Search for MDVRPTWSD

Multi-Depot vehicle routing problem with time windows under share depot is similar to the MDVRPTW except for one constraint. In MDVRPTW, it is required that each vehicle must start and end at the same depot. The new variant, MDVRPTWSD

in which a vehicle may or may not return to the same depot where it starts from. We used the MDVRPTW implementation described in Section 3.4 with this new variant change.

Chapter 4

Experimental Results

4.1 Introduction

This chapter provides the experimental details and the performance of the proposed system on well known VRPTW[35] and MDVRPTW[1] datasets.

4.2 VRPTW Dataset

The publicly available Solomon's benchmark dataset[35] was used. It has six different types of datasets R1, R2, C1, C2, RC1, and RC2 with customers size of 100. These data are generated in two different ways, randomly represented by problem sets R1 and R2 and clustered given as problem sets C1 and C2. A third set RC1 and RC2 is a mix of random and clustered structures in the problem set. The customer coordinates are always same in all the datasets, and only differs in the width of the time windows. Some of the datasets have very tight time windows, while others have a very wide time window. A summary of these datasets is in Table 4.1.

Instances	Customers	Capacity
R1-type	100	200
R2-type	100	1000
C1-type	100	200
C2-type	100	700
RC1-type	100	200
RC2-type	100	1000

Table 4.1: Dataset for the VRPTW experiment

4.3 Experimental Setup for VRPTW

All the experiments were implemented in Java for 30 runs per problem on an Intel(R) Core(TM) i5.2500 CPU @ 3.30GHz with 4GB RAM on Windows 2007. A number of experiments were performed as follows.

- A comparative study between known GAs and ALPSGA.
- Accuracy comparison among ALPSGA and non-GA approaches using weighted sum.
- A comparative study among weighted sum the multi-objective evaluation strategies.

4.3.1 Parameters

All the parameters were chosen empirically and given in Table 4.2. Further details of the determination of these parameters are found in Appendix A.

Parameters	value
Number of Runs	30
Replacement	Generational
Population	500
Generation	2000
Crossover Rate	0.8
Mutation Rate	0.2
Tournament type	Selection
Number of layers	5
Elite size	2
Tournament size	4
Age Gap	50
Aging scheme	Polynomial

Table 4.2: Parameters setting for ALPSGA experiment

4.3.2 Comparison of GAs and ALPSGA Using Weighted Sum Fitness Evaluation

Table 4.3 and 4.4 show the comparative results with best published result. Comparisons are only done if the total number of vehicles used in the solution are same. The bold faced values indicate the best result in 30 runs. From the table 4.3 and 4.4, it is seen that using weighted sum fitness proposed system and best published results performance is comparable. Proposed ALPSGA performed better in 20/29 instances if we consider the number of vehicles. If we consider total distance, ALPSGA performed better in 18/29 with the narrow time window and 13/27 for the wider time window.

Instance	Best known	ALPSGA	Difference in vehicle	Difference in distance
C101	10|828.94 [37]	✓	0	0
C102	10|828.94 [37]	✓	0	0
C103	10|828.06 [37]	✓	0	0
C104	10|824.78 [37]	✓	0	0
C105	10|828.94 [37]	✓	0	0
C106	10|828.94 [37]	✓	0	0
C107	10|828.94 [37]	✓	0	0
C108	10|828.94 [37]	✓	0	0
C109	10|828.94 [37]	✓	0	0
R101	19|1650.80 [46]	✓,1667.35	0	+16.55
R102	17|1486.12 [46]	✓,1490.71	0	+4.59
R103	13|1292.68 [49]	✓, 1315.25	0	+22.57
R104	9|1007.31 [49]	10, 1014.05	+1	+6.74
R105	14|1377.11 [49]	✓1413.72	0	+36.61
R106	12|1252.03 [24]	✓	0	0
R107	10|1104.06 [49]	✓1132.26	0	+28.2
R108	9|963.88 [49]	10, 1000.16	+1	+36.28
R109	11|1194.73 [37]	12, 1166.62	+1	-28.11
R110	10|1118.84 [37]	11,**1087.30**	+1	-31.54
R111	10|1096.73 [37]	11,**1078.75**	+1	-17.98
R112	9|982.14 [20]	10, 1000.09	+1	+17.95
RC101	14|1696.95 [20]	15, **1635.23**	+1	-61.72
RC102	12|1554.75 [20]	**12, 1524.97**	0	- 29.78
RC103	11|1261.67 [49]	✓1290.01	0	+28.34
RC104	10|1135.52 [49]	✓1164.89	0	+29.37
RC105	13|1629.44 [37]	14,**1554.40**	+1	-75.04
RC106	11|1424.73 [24]	12 **1400.19**	+1	-24.54
RC107	11|1230.48 [49]	✓	0	0
RC108	10|1139.82 [20]	✓, 1207.95	0	+68.13
Best solution		17/29	20/29	-18/29,+11/29

Table 4.3: Comparison of the ALPSGA with best published results using weighted sum

Instance	Best known	ALPSGA	Difference in vehicle	Difference in distance
C201	3591.56 [39]	✓	0	0
C202	3 591.56 [39]	✓	0	0
C203	3 591.17 [31]	✓	0	0
C204	3 590.60 [39]	✓	0	0
C205	3 588.88 [39]]	✓	0	0
C206	3 588.49 [39]	✓	0	0
C207	3 588.29 [31]	✓**588.03**	0	-0.26
C208	3 588.32 [31]	✓	0	0
R201	4 1252.37 [19]	✓1262.03	0	+9.66
R202	3 1191.70 [6]	✓1252.89	0.0	+61.19
R203	3 939.50 [19]	✓1013.94	0	+74.44
R204	3 828.78 [39]	**3,762.20**	0	-66.58
R205	3 994.43 [6]	✓1043.46	0	+49.03
R206	3 906.14 [31]	✓	0	0
R207	2 890.61 [6]	✓902.59	0	+ 11.98
R208	2 726.82 [39]	✓746.43	0	+19.61
R209	3 909.16 [39]	✓934.70	0	+25.54
R210	3 939.37 [6]	✓1000.06	0	+60.69
R211	2 891.11 [39]	3 **808.43**	+1	-82.68
RC201	4 1406.94 [32]	✓1413.51	0	+6.57
RC202	3 1365.64 [36]	4, **1165.11**	+1	-200.53
RC203	3 1049.62 [19]	✓1076.34	0	+26.72
RC204	3 798.46 [36]	✓**714.16**	0	-84.3
RC205	4 1297.65 [19]	✓1314.64	0	+16.99
RC206	3 1146.32 [6]	✓1169.41	0	+23.09
RC207	3 1061.14 [32]	✓1099.92	0	+38.78
RC208	3 828.24 [6]	✓855.40	0	+27.16
Best solution		13/27	25/27	-13/27,+14/27

Table 4.4: Comparison of the ALPSGA with best published results using weighted sum

4.3.3 Comparison of GAs and ALPSGA Using Pareto Fitness Evaluation

Table 4.5 and 4.6 show the comparative results with known GAs using multi-objective Pareto ranking. From the table, we could deduce that our proposed ALPSGA outperformed the Ombuki *et al.*[37] and Keivan *et al.* [14]. ALPSGA is best in 22 of 29 instances if we consider the total distance in Pareto ranking as our main objective, and 24 of 29 if considered the number of vehicles.

Instance	Ombuki *et al.*[37]	Keivan *et al.*[20]	ALPSGA
C101	**10,828.94**	**10,828.94**	✓
C102	**10,828.94**	**10,828.94**	✓
C103	**10,828.06**	**10,828.06**	✓
C104	**10,824.78**	**10,824.78**	✓
C105	**10,828.94**	**10,828.94**	✓
C106	**10,828.94**	**10,828.94**	✓
C107	**10,828.94**	**10,828.94**	✓
C108	**10,828.94**	**10,828.94**	✓
C109	**10,828.94**	**10,828.94**	✓
R101	19, 1690.28 20, 1664.13	19,1677 **20,1651**	**19 1667.35**
R102	17,1513.74 **18,1487.07**	18,1511 **19,1494.7**	**17,1496.4**
R103	14, 1237.05	14,1287 **15,1264.2**	**13,1315.67 14, 1224.95**
R104	10, 1020.87 11, 1010.24	**10,974.24**	10,1014.36
R105	**14,1415.13 15,1390.12**	15, 1424.6 16, 1382.5	**14, 1402.75** 15,1398.92
R106	13, 1254.22	13, 1270	**12,1273 13,1232.3**
R107	**11,1100.52**	11, 1108	**10,1132.26 11,1088.76**
R108	10, 975.34	10, 971	**10,960.26 9,1007.16**
R109	**12,1169.85**	12, 1212.3 14, 1206.7	**10,1252.8**
R110	11, 1112.21	12, 1156	**10,1181.2 11,1103.6**
R111	**11,1084.76 12,1079.8**	11, 1111.9	**10,1164.3**
R112	**10,976.99**	10,1036.9 **11,1011.5**	**9,991.4**
RC101	15, 1636.92	15,1690.6 **16,1678.9**	**14,1681.5 15, 1635.23**
RC102	**14,1488.36**	14, 1509.4 15, 1493.2	**13,1524.97** 14, 1503.85
RC103	**12,1306.42**	12, 1331.8	**11, 1290.01**
RC104	**10,1140.52**	11, 1177.2	10,1164.89
RC105	14, 1616.56 16, 1590.25	**15,1611.5 16,1589.4**	13, 1631.3
RC106	**12,1454.61** 13,1408.70	13,1437.6 **14,1425.3**	**12,1400.19 13,1388.87**
RC107	12, 1254.26	**11,1222.1**	11,1223.56 **12,1187.3**
RC108	**10,1141.34**	11, 1156.5	10,1207.95 11, 1141.96
Best solution	19/29	17/29	24/29, 22/29

Table 4.5: Comparison of the ALPSGA with known GAs using Pareto ranking

Instance	Ombuki *et al.*[37]	Keivan *et al.*[20]	ALPSGA
C201	**3,591.56**	**3,591.56**	✓
C202	**3,591.56**	**3,591.56**	✓
C203	**3,591.17**	**3,591.17**	✓
C204	**3,596.55**	3,599.96	✓
C205	**3,588.88**	**3,588.88**	✓
C206	**3,588.49**	3,588.88	✓
C207	3,588.29	3,591.56	✓**588.03**
C208	**3,588.32**	**3,588.32**	✓
R201	4, 1268.44 7, 1173.75	4,1351.4	**4,1232.21**
R202	4, 1112.52 5, 1046	**4,1091.22**	4 **1087.52 3 1252.89**
R203	**3,952.52**	3, 1041.0 6, 978.5	✓968.2
R204	**3,828.78**	3,1130.1 4,927.84 5,831.8 **6,826.3**	**2,900.92 3,762.20**
R205	**3,994.43** 5,954	4, 1087.8 3, 1422.3	**3,994.43**
R206	**3,919.73 4,889.39**	3,940.12	3,935.30
R207	**3,825.07 4,822.90**	2,890.61	3,833.59
R208	**2,773.13 3,719.17**	3,774.7	**2 746.43**
R209	3,971.70 5,874.95	4,1008	**3,934.70**
R210	3,985.38 5,930.42	**3,938.3**	3,1000.06
R211	**3,833.76 4,761.10**	4,1101 3,1310.4	**3, 808.43**
RC201	4,1423.73 7,1306.34	4 1423.2	**4,1406.94**
RC202	**4,1183.88 8,1118.05**	4 1369	**4,1183.88**
RC203	3,1131.78 5,951.08	4,1060 **6,1020.1**	**3,1049.62**
RC204	3,806.44 4,796.14	3 901	**3,788.16 4,796.14**
RC205	4,1352.39 **7,1181.86**	4 1410	**4,1297.65**
RC206	3,1269.64 5,1080.50	4 1194	**3,1146.32**
RC207	**3,1140.23 5,982.58**	**4,1040**	3,1161.14
RC208	3,881.20 4,785.93	3 898.4	**3,828.24**
Best solution	15/27	10/27	22/27

Table 4.6: Comparison of the ALPSGA with known GAs using Pareto ranking

In Table 4.7 and 4.8 show the comparative result with the published results. Comparisons are only done if the total number of vehicles used are the same. The column labeled "Best known" gives the best-known solutions, and column "Wsum" gives the best solution in 30 runs, where the VRPTW was considered as a single objective problem by using the weighted sum fitness evaluation technique. The column labeled "P vehicles" and "P distance" show the best solution in 30 runs, when VRPTW was considered as a multi-objective optimization problem and Pareto ranking technique was used as a fitness evaluation function in ALPSGA. P vehicles solution is the rank 1 solution that has the minimum number of vehicles, while P distance solution is the rank 1 solution with minimum travel distance. Bold faced values in table 4.7 and 4.8 indicate an improvement on the best currently known published results from the literature. Details of the best-known results are given in appendix A.

Instance	Best known	Wsum	P vehicles	P distance
C101	10|828.94 [37]	✓	✓	✓
C102	10|828.94 [37]	✓	✓	✓
C103	10|828.06 [37]	✓	✓	✓
C104	10|824.78 [37]	✓	✓	✓
C105	10|828.94 [37]	✓	✓	✓
C106	10|828.94 [37]	✓	✓	✓
C107	10|828.94 [37]	✓	✓	✓
C108	10|828.94 [37]	✓	✓	✓
C109	10|828.94 [37]	✓	✓	✓
R101	19|1650.80 [46]	19,1667.35	19,1667.35	19,1667.35
R102	17|1486.12 [46]	17,1490.71	17,1490.714	17,1490.71
R103	13|1292.68 [49]	13, 1315.25	13, 1315.25	**14, 1224.95**
R104	9|1007.31 [49]	10, 1014.05	10, 1014.05	10, 1014.056
R105	14|1377.11 [49]	✓1413.72	✓1413.72	✓1402.67
R106	12|1252.03 [24]	✓	✓	13 **1232.3**
R107	10|1104.06 [49]	✓1132.26	11 **1088.76**	11 **1088.76**
R108	9|963.88 [49]	10, 1000.16	✓1007.01	9, 1000.16
R109	11|1194.73 [37]	12, 1166.62	12, 1166.62	12, 1166.622
R110	10|1118.84 [37]	11,**1087.30**	11,**1087.30**	11,**1087.30**
R111	10|1096.73 [37]	11,**1078.75**	✓1164.3	✓1167.2
R112	9|982.14 [20]	10, 1000.09	✓991.4	✓1023.3
RC101	14|1696.95 [37]	15, **1635.23**	15, **1635.23**	15, **1635.23**
RC102	12|1554.75 [37]	12 **1524.97**	14, **1503.85**	14, **1503.85**
RC103	11|1261.67 [49]	✓1290.01	12, 1314.86	12, 1314.86
RC104	10|1135.52 [49]	✓1164.89	11, 1177.98	11, 1177.98
RC105	13|1629.44 [37]	14,**1554.40**	14,**1554.40**	14,**1554.40**
RC106	11|1424.73 [24]	12 1400.19	13, **1388.87**	13, **1388.874**
RC107	11|1230.48 [49]	✓	12, 1268.46	12, 1268.46
RC108	10|1139.82 [20]	10, 1207.95	11, 1141.96	11, 1141.96
Best solution		16/29	14/29	17/29

Table 4.7: Solomon Benchmark with narrow time windows;comparison of our ALPSGA with best published result

Instance	Best known	Wsum	P vehicles	P distance
C201	3591.56 [39]	✓	✓	✓
C202	3 591.56 [39]	✓	✓	✓
C203	3 591.17 [39]	✓	✓	✓
C204	3 590.60 [39]	✓	✓	✓
C205	3 588.88 [39]	✓	✓	✓
C206	3 588.49 [39]	✓	✓	✓
C207	3 588.29 [39]	✓**588.03**	✓**588.03**	✓**588.03**
C208	3 588.32 [31]	✓	✓	✓
R201	4 1252.37 [19]	✓1262.03	✓1262.03	✓1262.03
R202	3 1191.70 [6]	✓1252.89	✓**1087.52**	4, **1093.00**
R203	3 939.50 [19]	3,1013.94	3,1013.94	3,1013.94
R204	3 828.78 [19]	**3,762.20**	**2, 900.29**	**3,762.20**
R205	3 994.43 [6]	3, 1043.46	3, 1043.46	3, 1043.46
R206	3 906.14 [31]	✓	✓	✓
R207	2 890.61 [6]	✓902.59	✓902.59	✓934.61
R208	2 726.82 [39]	2 746.43	2 746.43	2 746.43
R209	3 909.16 [39]	3 934.70	3 934.70	3 934.70
R210	3 939.37 [6]	3, 1000.06	2, 1086.34	3, 1000.06
R211	2 891.11 [39]	3 **808.43**	3 910.09	3 **808.43**
RC201	4 1406.94 [32]	4, 1413.51	4, 1413.51	4, 1413.51
RC202	3 1365.64 [**?**]	4, **1165.11**	5, **1215**	4, **1165.11**
RC203	3 1049.62 [19]	3, 1076.34	3, 1076.34	3, 1076.34
RC204	3 798.46 [36]	✓**714.16**	✓**788.16**	✓**714.16**
RC205	4 1297.65 [19]	4, 1314.64	4, 1314.64	4, 1314.64
RC206	3 1146.32 [6]	3, 1169.41	3, 1169.41	3, 1169.41
RC207	3 1061.14 [32]	3, 1099.92	3, 1099.92	3, 1099.92
RC208	3 828.24 [6]	3, 855.40	3, 1099.92	3, 1099.92
Best solution		13/27	16/27	16/27

Table 4.8: Solomon Benchmark with wide time windows;comparison of our ALPSGA with best published result

Table 4.9, 4.10, and 4.11 report the ALPSGA results for Solomon's benchmark, showing the average and the best distances. For 34 out of 56 instances, ALPSGA for the VRPTW manages to find the best-known solution within 30 runs, whereas 21 out of 56 instances match the average of 30 runs. The subclass with clustered customers is very efficiently optimized (see table 4.9), where the average is matched with the best solution for all instances.

Instance	Average	Best	Instance	Best	Average
c101	**828.94**	**828.94**	C201	**591.56**	**591.56**
c102	**828.94**	**828.94**	C202	**591.56**	**591.56**
c103	**828.94**	**828.94**	C203	**591.17**	**591.17**
c104	**828.94**	**828.94**	C204	**590.60**	**590.60**
c105	**828.94**	**828.94**	C205	**588.88**	**588.88**
c106	**828.94**	**828.94**	C206	**588.49**	**588.49**
c107	**828.94**	**828.94**	C207	**588.29**	**588.29**
c108	**828.94**	**828.94**	C208	**588.32**	**588.32**
c109	**828.94**	**828.94**			

Table 4.9: Average and the minimum total travel distance with clustered customers (in bold faced if best match or outperform published results

Instance	Average	Best	Instance	Average	Best
r101	1671.97	1667.35	r201	1265.02	1262.03
r102	1496.87	1490.71	r202	1103.78	1093.00
r103	1323.95	1315.67	r203	1016.62	1013.94
r104	**1014.05**	**1014.05**	r204	833.09	762.20
r105	**1402.75**	**1402.75**	r205	1047.91	1043.59
r106	1255.59	**1259.90**	r206	936.51	935.30
r107	1108.28	**1088.76**	r207	835.90	**833.59**
r108	1014.23	1000.06	r208	748.02	746.43
r109	1195.27	**1166.62**	r209	939.37	934.70
r110	**1118.84**	**1118.84**	r210	1002.87	1000.06
r111	1096.73	**1078.75**	r211	892.62	**808.43**
r112	1012.65	1000.09			

Table 4.10: Average and the minimum total travel distance with random customers (in bold faced if best match or outperform published results

Instance	Average	Best	Instance	Average	Best
rc101	1686.95	**1635.23**	rc201	148.49	1413.51
rc102	1554.75	**1503.85**	rc202	1186.93	**1165.11**
rc103	1291.95	1290.01	rc203	1083.26	**1176.11**
rc104	1165.52	1164.89	rc204	834.92	814.90
rc105	**1554.40**	**1554.40**	rc205	1328.37	1314.64
rc106	1424.73	**1388.87**	rc206	1174.85	1169.41
rc107	1271.52	1268.48	rc207	1103.67	1099.92
rc108	1223.34	1207.95	rc208	863.68	855.40

Table 4.11: Average and the minimum total travel distance with random and clustered customers (in bold faced if best match or outperform published results)

Table 4.12 illustrates that our proposed ALPSGA obtained better or similar average number of vehicles as compared to some of the well known published GA-based methods for VRPTW.

Methods →	Ombuki at al.	Keivan at al.	Nalpa at al.	ALPSGA	ALPSGA-Pareto
C1	**10.0**	**10.0**	**10.0**	**10.0**	**10.0**
C2	**3.0**	**3.0**	**3.0**	**3.0**	**3.0**
R1	12.7	13.1	**11.92**	**2.73**	**11.50**
R2	3.2	3.2	**2.73**	3.2	2.8
RC1	12.3	12.3	**11.50**	12.3	12.3
RC2	3.4	3.3	**3.25**	**3.25**	**3.25**

Table 4.12: Comparison of average number of vehicles on Solomon's VRPTW instances with other published results

In figures 4.1-3, we show some of the network typologies obtained using our proposed system. Fig 4.1 represent a dataset c101, where customers are clustered together and have a narrow time window. Fig 4.2 shows the network for r201, where all the customers are randomly dispersed but have a wider time window. Fig 4.3 illustrates the network of rc101 instance, where customers are randomly as well as clustered and have a narrow time window.

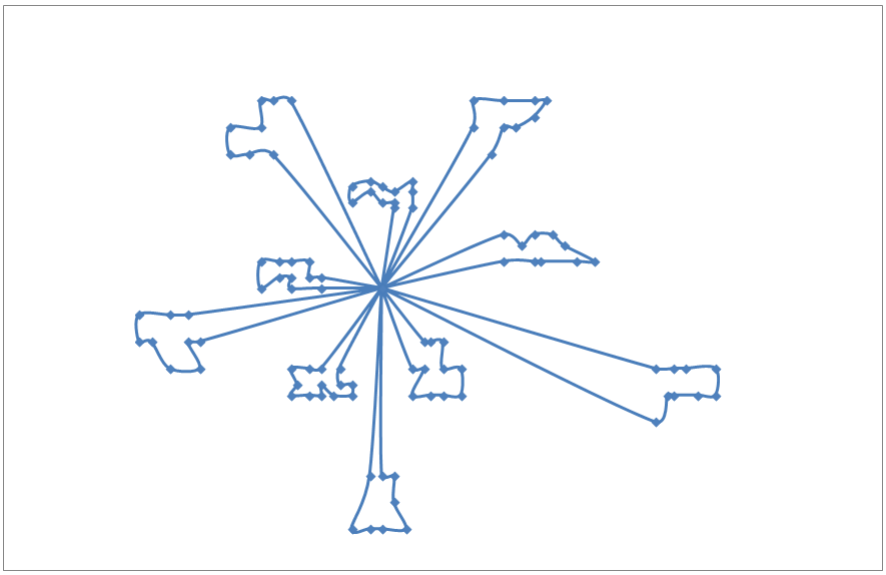

Figure 4.1: Network topology for 100 geographically clustered customers with a narrow time window for instance c101

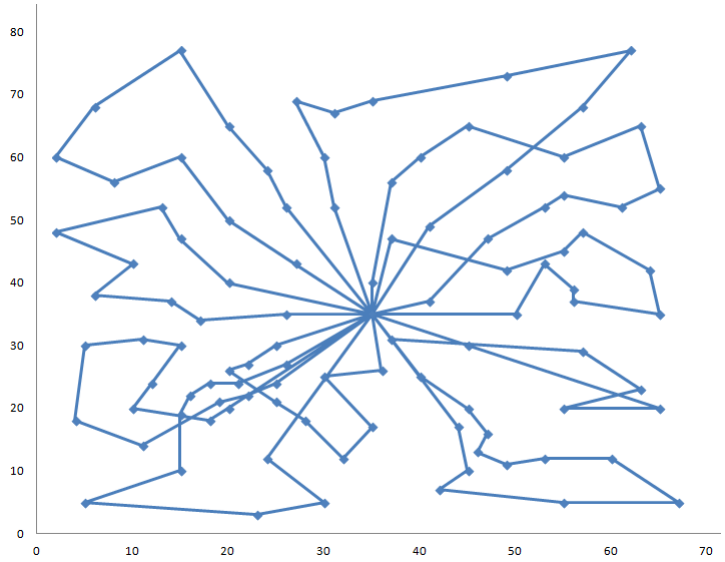

Figure 4.2: Network topology for 100 geographically clustered customers with a wide time window for instance r201

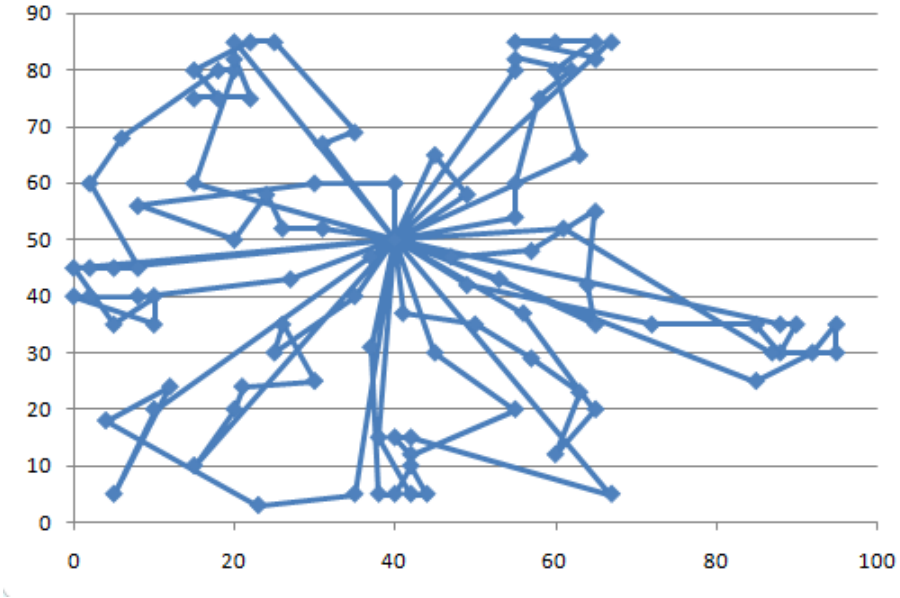

Figure 4.3: Network topology for 100 geographically clustered customers with a wide time window for instance rc101

Instances	Customers	Capacity	Route duration	Depots	Vehicles
pr01	48	200	500	4	2
pr02	96	196	480	4	3
pr03	144	185	440	4	5
pr04	192	180	420	4	6
pr05	240	175	400	4	7
pr06	288	200	500	6	2
pr07	72	200	500	6	2
pr08	144	190	475	6	3
pr09	216	180	450	6	4
pr10	288	170	425	6	5
pr11	48	200	500	4	1
pr12	96	195	480	4	2
pr13	144	190	460	4	3
pr14	192	185	440	4	4
pr15	240	184	420	4	5
pr16	288	175	400	4	6
pr17	72	200	500	6	1
pr18	144	190	275	6	2
pr19	216	180	450	6	3
pr20	288	170	425	6	4

Table 4.13: Dataset for the MDVRPTW experiment

4.4 MDVRPTW Dataset

A well-known dataset for MDVRPTW designed by Cordeau(2001) used in this thesis which is available in [35]. These datasets made up of 20 instances. First 10 instances pr01-pr10 have narrow time windows, while the next 10 instances pr11-pr20 have a wider time window. pr01-pr20 have 48-244 customers and 4-6 depots. Characteristics of MDVRPTW instances are shown in Table 4.13.

4.5 Experimental Setup for MDVRPTW

All the experiment was implemented in Java with 10 runs for each data set on Intel(R) Core(TM) i5.2500 CPU @ 3.30GHz with 4GB RAM on Windows 2007. The number of experiments performs is given below. Previous research on MDVRPTW was tried to reduce the total distance only and to the best of our knowledge, there is no multi-objective approach on this variant of the VRP problem. For the MDVRPTW best result available by Courdue[22] and in the recent years some of the datasets have been improved[13, 38].

Parameters	value
Number of Runs	30
Population	500
Generation	2000
Crossover Rate	0.8
Mutation Rate	0.2
Tournament type	Selection
Tournament size	4

Table 4.14: Parameters setting for GA experiment

Parameters	value
Replacement	Generational
Aging scheme	Polynomial
Layer replacement	Best of two layers

Table 4.15: Parameters setting for ALPSGA experiment

4.5.1 Parameters

The final evolutionary parameters used to set up GA are specified in Table 4.14. Additional parameter settings used in the ALPSGA systems are listed in Table 4.15. The main reason why we choose these parameters is given in [53]. Our proposed system chooses some of the parameters empirically. We show how we determined the given parameters empirically given in Appendix A.

Instances	Vehicles available	GA	difference in vehicle
pr01	8	1182.6 **5**	-3
pr02	12	1900.88 **10**	-2
pr03	16	2683.08 **12**	-4
pr04	20	3282.74 **17**	-3
pr05	24	3604.02 **21**	-3
pr06	28	4179.99 **25**	-3
pr07	12	1605.96 **8**	-4
pr08	18	2277.50 **4**	-4
pr09	24	3283.40 **17**	-7
pr10	30	4198.93 **25**	-5
pr11	4	970.57 5	+1
pr12	8	1603.69 **8**	0
pr13	12	2255.38 **12**	0
pr14	16	2470.77 15	+1
pr15	20	2833.28 **21**	0
pr16	24	3322.50 23	+1
pr17	6	1303.4 7	+1
pr18	12	1883.92 13	+1
pr19	18	2913.64 **16**	-2
pr20	24	3332.08 **25***	0
Best solution			15/20

Table 4.16: Best result obtained by proposed GA on Cordue's dataset of MDVRPTW using weighted sum procedure

4.5.2 Results

Table 4.16 shows the result of MDVRPTW using weighted sum technique with proposed GA. From the table 4.16, it is seen that our GA reduced the number of vehicles in 15 out of 20 instances and in 4 instances it is used the same number of vehicles available. We got one infeasible solution marked with a "*" sign (see pr20 in Table 4.16) where GA found a solution using more than the upper bound of the vehicles.

Instances	GA	ALPSGA	Difference in distance	Difference in vehicle
pr01	1182.65	**1182.02 5**	-0.58	0
pr05	3604.02 21	**3534.3 22**	-69.72	0
pr09	3283.40 17	**3201.40 18**	-82.0	0
pr10	4198.26 25	**4072.3 26**	125.96	0
pr11	970.5 5	**943.4 5**	-27.1	0
pr15	2833.28 21	**2734.8 21**	-98.48	0
pr16	3322.50 23	**3311.4 23**	-11.1	0
pr17	1303.4 7	**1303.4 7**	-	0
pr18	1883.92 13	**1826.2 13**	-57.72	0
pr19	2913.64 16	**2901.64 16**	-12	0
pr20	3332.08 25*	**3327.08 25***	-5	0
Best solution		11/11	11/11	11/11

Table 4.17: Comparison of the non-ALPS based GA with ALPSGA using weighted sum

4.5.3 Comparison between GA and ALPS-GA

From Table 4.17 and 4.18 we can say that ALPSGA is performed better compare to the GA. The comparison is only done if the number of vehicles used is same. From Table 4.17 we can deduce that the ALPSGA performed better than the GA. In 10 out of 11 instances, it reduced the distance and the number of vehicles remains constant in all the instances.

Instances	GA	ALPSGA-vehicles	ALPSGA-distance
pr01	1088.898	**1086.82 8**	**1086.82 8**
pr02	1900.88 10	**1886 10**	**1871.23** 11
pr03	2683.08 12	**2625** 13	**2572.19** 14
pr07	1605.96 8	**1488** 10	**1471.52** 11
pr09	3130.30 20	**3041.63 20**	**3041.63 20**
pr11	982.82 5	**955.45 5**	**955.45 5**
pr15	2704.4 21	**2623.6 21**	**2693 20**
pr17	1262.4 8	**1233.87 8**	**1233.87 8**
pr19	2913.64 16	**2522 16**	**2465** 17
pr20	3414.9 25*	**3285 25***	**3285 25***
Best solution		8/10	10/10

Table 4.18: Comparison of the non-ALPS based GA with ALPSGA using Parato ranking

Instances	ALPSGA	ALPSGAm	Difference in distance	Difference in vehicle
pr01	1182.025	**1182.00 5**	-0.02	0
pr05	3534.3 21	**3512.83 21**	-21.47	0
pr07	1602.40 8	**1572.45 8**	-29.95	0
pr08	2277.50 14	**2229.4 14**	-48.1	0
pr11	943.4 5	**943.21 5**	-0.19	0
pr13	2255.38 12	**2168.34 12**	-87.04	0
pr14	2470.77 15	**2449.15 15**	-21.62	0
pr15	2734.8 21	**2588.34 20**	-148.46	-1
pr16	3311.4 23	3345.11 23	+22.4	0
pr17	1303.4 7	**1217.47 7**	-85.93	0
pr18	1826.2 13	**1817.3 13**	-8.9	0
pr19	2901.64 16	**2570.2 16**	-456.56	0
pr20	3327.08 25*	3298.83 25*	-28.25	0
Best solution		10/13		

Table 4.19: Comparison of the GA with neighborhood and global mutation technique using weighted sum

Instances	ALPSGA-vehicles	ALPSGA-distance	ALPSGAm-vehicles	ALPSGAm-distance
pr01	**1086.82 8**	**1086.82 8**	**5** 1193.48	**8 1086.82**
pr02	1886.96 10	1871.23 11	**10 1886.96**	**11 1871.23**
pr03	**2625** 13	**2572.19** 14	**13 2625.30**	**14 2572.19**
pr07	1488.48 10	**1471** 11	**9** 1531.12	11 1491.17
pr09	3041.4 20	3041.4 20	**18** 3238.8	**20 3041.35**
pr11	955.45 5	955.45 5	**5 950.67**	**5 950.67**
pr15	**2623.6 21**	2693.7 20	21 2685.74	**21 2685.74**
pr17	**1233 8**	**1233 8**	**7 1235.70**	8 1256.56
pr19	**2522 16**	**2465** 17	17 2549.51	17 2549.51
pr20	**3312.8 25***	**3285 25***	25 3496.5	25 3496.5
Best solution	4/10	6/10	7/10	7/10

Table 4.20: Comparison of the GA with neighborhood and global mutation technique using Pareto ranking

From the table 4.19 and 4.20, it is clear that neighborhood and global mutation performed well. It outperformed 13 instances and reduced the number of vehicles in two instances. We compare our result with ALPSGA after incorporating the neighborhood and global mutation we developed for MDDVRPTW. From the table 4.19, it is seen that with the help of the mutation, results improved slightly. New results outperformed in all 7 instances, moreover it also able to reduce the number of vehicles in one instance.

Instances	wALPSGA	Distance pALPSGA	vehicles pALPSGA
pr01	1222.83	1182.6	1206.27
pr02	1941.11	1900.88	1924.55
pr03	2723.31	2683.08	2706.75
pr04	3322.97	3282.74	3306.41
pr05	3644.25	3604.02	3627.69
pr06	4220.22	4179.99	4203.66
pr07	1646.19	1605.96	1629.63
pr08	2317.73	2277.5	2301.17
pr09	3323.63	3283.4	3307.07
pr10	4239.16	4198.93	4222.6
pr11	1010.8	970.57	994.24
pr12	1643.92	1603.69	1627.36
pr13	2295.61	2255.38	2279.05
pr14	2511	2470.77	2494.44
pr15	2873.51	2833.28	2856.95
pr16	3362.73	3322.5	3346.17
pr17	1343.63	1303.4	1327.07
pr18	1924.15	1883.92	1907.59
pr19	2953.87	2913.64	2937.31
pr20	3372.31	3332.08	3355.75

Table 4.21: Average of 30 runs using ALPSGA on Courdeau's MDVRPTW instances

Table 4.21 gives the average performance of the ALPSGA. The columns labeled "wALPSGA" gives the average of 30 runs using the weighted sum fitness function. For the Pareto experiments, two columns labeled "pALPSGA" gives the average of all the rank 1 solutions in 30 runs for distance and vehicles respectively. By comparing the average result with the Table 4.19 and 4.20, we can deduce that the average performance of proposed ALPSGA is good.

In figures 4.4 and 4.5, we show some of the network solutions we obtained using ALPSGA. Figure 4.4 represent a dataset pr03, which consists of 144 customers randomly dispersed with the narrow time window. Our solution got a result with 4 depots. Figure 4.5 illustrates another dataset pr11 with 48 customers. This dataset has a wider time window; hence the vehicles can serve more customers if vehicle capacity permits.

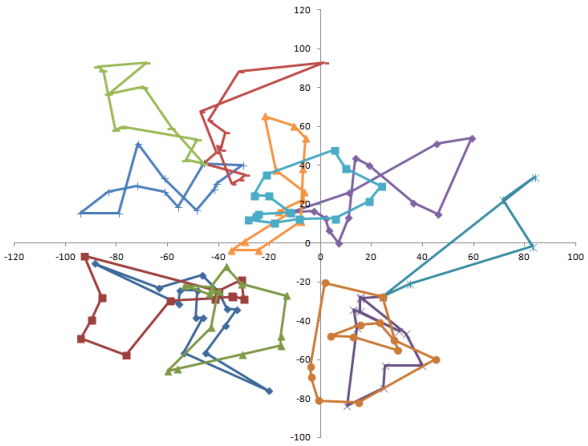

Figure 4.4: Network topology for 144 geographically clustered customers for instance pr03

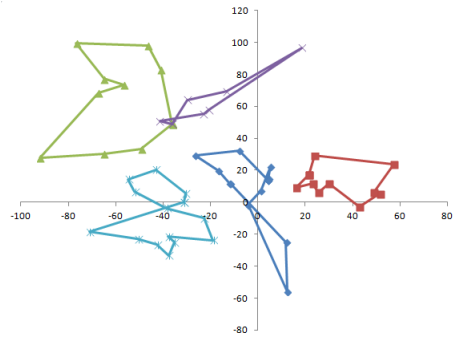

Figure 4.5: Network topology for 48 geographically clustered customers for instance pr11

In Figure 4.6, shows the fitness plot of 5 layers. Layer 0 showing spiky fitness curve whereas layer 4 showing a smooth convergence over the generations. Consistent spikes of layer 0 fitness curve occur due to the introduction of new individuals at regular interval of time. The transfer of individuals at each layer except the final layer seems to happen at a different time because of the age gap and different aging scheme. Figure 4.7 shows the last layer fitness plot for pr10 instance. It shows that the fitness is improving with the generations. Though the slope is not smooth, it kept improving with the inter-layer transfer.

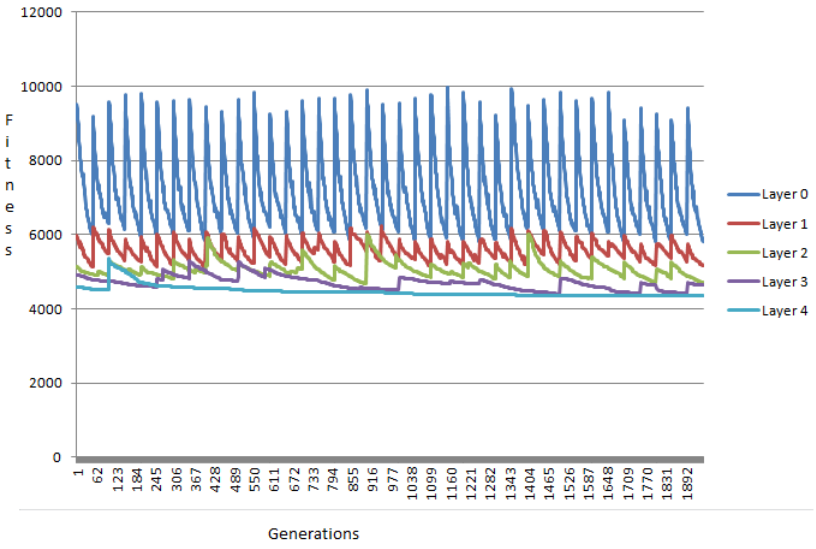

Figure 4.6: Fitness plot for pr11 with 5 layers

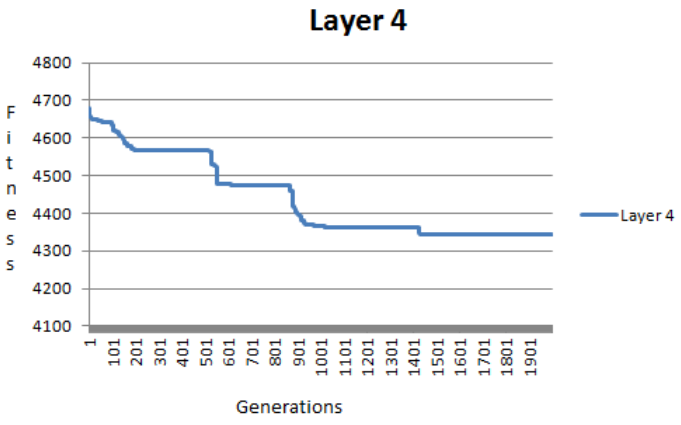

Figure 4.7: Fitness plot for pr10 in layer 4

Clustering of geographically dispersed customers

In order get a solution from a set of geographically dispersed customers, we first assigned customers to the nearest depot and finally optimized route formed using the evolutionary process. We have used pr03 dataset to illustrate the process. Fig 4.8 shows the dispersed customers, and fig 4.9 shows the clusters customers to the nearest depot. Table 4.22 shows the optimized route from ALPSGA.

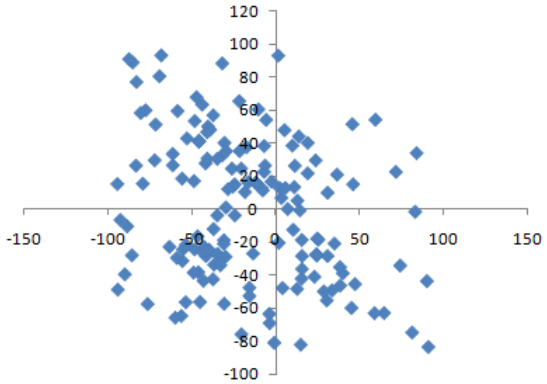

Figure 4.8: Geographically dispersed customers

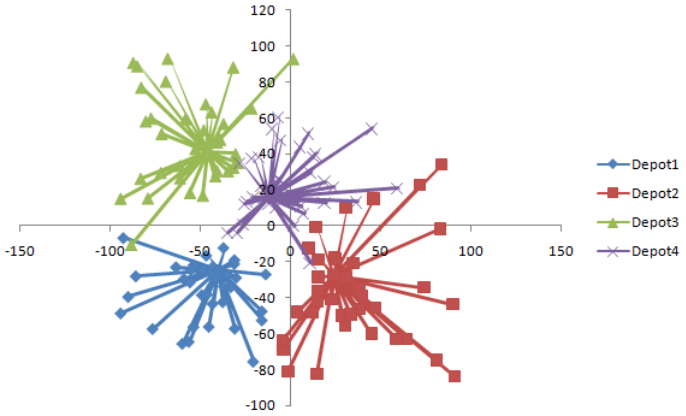

Figure 4.9: Customers assigned to the nearest depot

Routes	Ordered customers list
Depot 1	114-110-119-69-40-132-3-80-17-45-91-87-107
Depot 1	95-62-133-129-73-127-121-50-112-12-10-88-52-14
Depot 1	39-11-78-109-26-139-13-1-71
Depot 2	117-8-125-101-79-19-70-135-105-34-100-72-74-141-24
Depot 2	55-67-85-57-36-61-111-83
Depot 2	37-51-46-68-6-4-56-115
Depot 2	75-126-54-98-49
Depot 3	134-81-122-23-102-48-97-64-137-113-47-131-142
Depot 3	118-42-38-20-103-27-31-44-144-99-130
Depot 3	124-2-136-60-140-89-9-77-30-32-94
Depot 4	25-29-76-116-120-128-108-58-22-18-16
Depot 4	43-63-143-15-41-138-96-66-7-90-53-82-59-106
Depot 4	92-86-21-65-93-33-123-104-84-28-35-5

Table 4.22: The optimised route from ALPSGA for Pr03 dataset

Instances	ALPSGA-MDVRPTW	ALPSGA-MDVRPTWSD	Difference in vehicle	Difference in distance
pr01	1086.82 8 1193.48 5	**1083.52 8**	0	-5.37
pr02	1871.23 11 1886.96 10	**1868.3 11**	0	-3.64
pr03	2572.25 14 2625.30 13	**2468.84 14**	0	-4.41
pr07	1471.39 11 1488.48 10	**1466 11**	0	-5.89
pr11	950.67 5	943.47 5	0	-7.2
pr12	1609.54 9	1609.54 9	0	0
pr13	2181.91 12	2182.3 12	0	+0.39
pr17	1233.76 8	1233.76 8	0	0
pr18	1879.76 12	**1858.21 12**	0	-21.55
Best solution		5/9	9/9	

Table 4.23: Comparison of MDVRPTW and MDVRPTW with share depot using Pareto ranking

Instances	ALPSGA-MDVRPTW	ALPSGA-MDVRPTWSD	Difference in distance	Difference in vehicle
pr01	1182.00 5	1182.00 5	0	0
pr05	3512.83 21	**3489.61 20**	-23.22	-1
pr07	1572.45 8	**1563.78 8**	-8.67	0
pr08	2329.4 14	2329.4 14	0	0
pr11	943.21 5	**984.87 4**	+41.66	-1
pr13	2168.34 12	**2002.12 12**	-166.22	0
pr14	2449.15 15	2449.15 15	0	0
pr15	2588.34 20	2588.34 20	0	0
pr16	3345.11 23	3345.11 23	0	0
pr17	1217.47 7	**1236.24 6**	+18.77	-1
pr18	1817.3 13	**1789.27 12**	-28.03	-1
Best solution		6/11	9/11	11/11

Table 4.24: Comparison of MDVRPTW and MDVRPTW with share depot using weighted sum

4.6 MDVRPTW Dataset where vehicle return to nearest depot

No benchmark date set is available for MDVRPTW where vehicles return to the nearest depot, as it is a new variant of MDVRPTW. However, the main differences between this approach and MDVRPTW are the constraints of the depot ending, and the other basic data is similar. There is benchmark data available for MDVRPTW which we used in section 4.2, and it is available at [6].

From the table 5.23, it is seen that with share depot constrain total distance of the solution reduced slightly. The number of vehicles used in both solution remains same. Out of 9 instances, it is best in 5 instances, and in all other cases, it gives exactly same result obtained with MDVRPTW.

Table 5.24 shows that using the share depot constraint and weighted sum fitness function overall performance is better. A comparison was done when the number of vehicles are equal or less with the new constraint. Out of 11 instances, it is best

in 6 data-sets. If we consider only the number of vehicle in the solution then the share depot constraint outperforms all the instances. In case of total distance, it outperforms 9 out of 11 instances. With the new constraint it able to reduce both the total distance and number of vehicles in pr05 and pr18 data-sets.

Chapter 5

Conclusion

5.1 Summary

This thesis presents an investigation of different variants of vehicle routing problem with time windows. We investigated multi-depot vehicle routing problem with time windows using fixed destination as well as share depot destination. Multi-population based genetic algorithm is applied to all these variants of VRP, and the result gives better performance than the standard genetic algorithm.

The vehicle routing problem has been studied for over fifty years now. Many variants have been considered, and the one got more attention is vehicle routing problem with time windows because it is related more to real life problems like bin packing, waste collection, newspaper delivery or any other scheduling problem. So far a good exact method to solve VRP has not yet been found. Thus, researchers have to rely on different heuristics and meta-heuristics for the VRP, which currently produce more optimal solutions.

In this thesis, we used the genetic algorithm (GA), which proven to be good to produce a better result for the VRPTW and other variants of VRP. Several GAs applied to this problem, but only a couple of research has been done where VRPTW problem is considered as a multi-objective problem. To the best of our knowledge, no research has been done on multi-depot VRPTW where the problem is considered as a multi-objective problem. Multi-depot vehicle routing problem with time windows under share depot resource (MDVRPTWSD) is another variant where no research is done by considering at as a multi-objective problem. We considered all these variants of VRPTW as a multi-objective problem and proposed a multi-population based GA. Different mutation technique employed to improved the overall result. Different multi-objective techniques are considered for comparison. The experimental results are very

competitive and produce a better result in many instances. Comparing with classical MDVRPTW problem with multi-objective MDVRPTW and MDVRPTWSD may reduce the total transportation cost significantly. Most importantly, multi-objective interpretation of all the variants of VRPTW gives us the solution without bias towards the number of the vehicle.

This thesis considered VRPTW, MDVRPTW, and MDVRPTWSD as a multi-objective optimization problem. To the best of our knowledge, this thesis first introduced multi-objective optimization for MDVRPTW and MDVRPTWSD problem. A multi-layers population-based GA is employed for all these variants of VRP.

5.2 Future Work

Below are some considerations that can be made for the future work:

- To work on a more complex variant, such as MDVRPTW under dynamic traffic environment or real-world based datasets.
- To try other clustering methods than the one applied in this thesis for MDVRPTW and MSVRPTWSD
- Other multi-objective fitness technique could be applied to the problem
- ALPS can be tried on other EAs for different variants of VRPTW

Bibliography

[1] `http://www.bernabe.dorronsoro.es/vrp/`. Accessed: Dec 16, 2016.

[2] Anthony Awuley and Brian J Ross. Feature selection and classification using age layered population structure genetic programming. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pages 2417–2426. IEEE, 2016.

[3] MO Ball, TL Magnanti, CL Monma, and GL Nemhauser. Network routing, handbooks in operations research and management, 1995.

[4] Walter J Bell, Louis M Dalberto, Marshall L Fisher, Arnold J Greenfield, Ramchandran Jaikumar, Pradeep Kedia, Robert G Mack, and Paul J Prutzman. Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer. *Interfaces*, 13(6):4–23, 1983.

[5] Olli Bräysy and Michel Gendreau. Vehicle routing problem with time windows, part i: Route construction and local search algorithms. *Transportation science*, 39(1):104–118, 2005.

[6] Olli Bräysy and Michel Gendreau. Vehicle routing problem with time windows, part ii: Metaheuristics. *Transportation science*, 39(1):119–139, 2005.

[7] Leo Budin, Marin Golub, and Andrea Budin. Traditional techniques of genetic algorithms applied to floating-point chromosome representations. *sign*, 1(11):52, 2010.

[8] Wen-Chyuan Chiang and Robert A Russell. Simulated annealing metaheuristics for the vehicle routing problem with time windows. *Annals of Operations Research*, 63(1):3–27, 1996.

[9] Wen-Chyuan Chiang and Robert A Russell. A reactive tabu search metaheuristic for the vehicle routing problem with time windows. *INFORMS Journal on computing*, 9(4):417–430, 1997.

[10] J-F Cordeau, M. Gendreau, G. Laporte, J-Y Potvin, and F. Semet. A guide to vehicle routing heuristics. *Journal of the Operational Research Society*, 53(5):512–522, 2002.

[11] Jean-François Cordeau, Gilbert Laporte, and Anne Mercier. A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational research society*, 52(8):928–936, 2001.

[12] Jean-François Cordeau, Gilbert Laporte, and Anne Mercier. Improved tabu search algorithm for the handling of route duration constraints in vehicle routing problems with time windows. *Journal of the Operational Research Society*, 55(5):542–546, 2004.

[13] Jean-François Cordeau and Mirko Maischberger. A parallel iterated tabu search heuristic for vehicle routing problems. *Computers & Operations Research*, 39(9):2033–2050, 2012.

[14] Jean-François Cordeau and Mirko Maischberger. A parallel iterated tabu search heuristic for vehicle routing problems. *Computers & Operations Research*, 39(9):2033–2050, 2012.

[15] George B Dantzig and John H Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

[16] Kenneth Alan De Jong. *Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, 1975.

[17] Luca Maria Gambardella, Éric Taillard, and Giovanni Agazzi. Macs-vrptw: A multiple ant colony system for vehicle routing problems with time window. *New Ideas in Optimization*, pages 63–76, 1999.

[18] Hermann Gehring and Jörg Homberger. Parallelization of a two-phase metaheuristic for routing problems with time windows. *Journal of heuristics*, 8(3):251–276, 2002.

[19] Michel Gendreau, Jean-Yves Potvin, Olli Bräumlaysy, Geir Hasle, and Arne Løkketangen. Metaheuristics for the vehicle routing problem and its extensions: A categorized bibliography. In *The vehicle routing problem: latest advances and new challenges*, pages 143–169. Springer, 2008.

[20] Keivan Ghoseiri and Seyed Farid Ghannadpour. Multi-objective vehicle routing problem with time windows using goal programming and genetic algorithm. *Applied Soft Computing*, 10(4):1096–1107, 2010.

[21] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[22] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. Cambridge, Massachusetts, second edition,MIT press, 1992.

[23] Gregory S Hornby. Alps: the age-layered population structure for reducing the problem of premature convergence. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 815–822. ACM, 2006.

[24] Gregory S Hornby. The age-layered population structure(alps) evolutionary algorithm. pages 8–12, 2009.

[25] Gregory S Hornby. A steady-state version of the age-layered population structure ea. In *Genetic Programming Theory and Practice VII*, pages 87–102. Springer, 2010.

[26] Jian Jun Hu and Erik D Goodman. The hierarchical fair competition (hfc) model for parallel evolutionary algorithms. In *Evolutionary Computation, 2002. CEC'02. Proceedings of the 2002 Congress on*, volume 1, pages 49–54. IEEE, 2002.

[27] Jianjun Hu, Erik D Goodman, Kisung Seo, and Min Pei. Adaptive hierarchical fair competition (ahfc) model for parallel evolutionary algorithms. In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, pages 772–779. Morgan Kaufmann Publishers Inc., 2002.

[28] Angel A Juan, Javier Faulin, Albert Ferrer, Helena R Lourenço, and Barry Barrios. Mirha: multi-start biased randomization of heuristics with adaptive local search for solving non-smooth routing problems. *Top*, 21(1):109–132, 2013.

[29] Sašo Karakatič and Vili Podgorelec. A survey of genetic algorithms for solving multi depot vehicle routing problem. *Applied Soft Computing*, 27:519–532, 2015.

[30] Niklas Kohl. *Exact methods for time constrained routing and related scheduling problems*. PhD thesis, Ph.D. thesis, Institute of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, 1995.

[31] Gilbert Laporte, Michel Gendreau, J-Y Potvin, and Frédéric Semet. Classical and modern heuristics for the vehicle routing problem. *International transactions in operational research*, 7(4-5):285–300, 2000.

[32] Jian Li, Yang Li, and Panos M Pardalos. Multi-depot vehicle routing problem with time windows under shared depot resources. *Journal of Combinatorial Optimization*, 31(2):515–532, 2016.

[33] R Garey Michael and S Johnson David. Computers and intractability: a guide to the theory of np-completeness. *WH Free. Co., San Fr*, pages 90–91, 1979.

[34] Mohammad Naeem. Using genetic algorithms for the single allocation hub location problem. Master's thesis, Department of Computer Science, Brock University, 2009.

[35] Jakub Nalepa and Miroslaw Blocho. Adaptive memetic algorithm for minimizing distance in the vehicle routing problem with time windows. *Soft Computing*, 20(6):2309–2327, 2016.

[36] Siamak Noori and S Farid Ghannadpour. High-level relay hybrid metaheuristic method for multi-depot vehicle routing problem with time windows. *Journal of Mathematical Modelling and Algorithms*, 11(2):159–179, 2012.

[37] Beatrice Ombuki, Brian J Ross, and Franklin Hanshar. Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, 24(1):17–30, 2006.

[38] Audrey Opoku-Amankwaah. Using age layered population structure for the multi-depot vehicle routing problem. Master's thesis, Department of Computer Science, Brock University, 2016.

[39] David Pisinger and Stefan Ropke. A general heuristic for vehicle routing problems. *Computers & operations research*, 34(8):2403–2435, 2007.

[40] Michael Polacek, Siegfried Benkner, Karl F Doerner, and Richard F Hartl. A cooperative and adaptive variable neighborhood search for the multi depot vehicle routing problem with time windows. *BuR-Business Research*, 1(2):207–218, 2008.

[41] Michael Polacek, Richard F Hartl, Karl Doerner, and Marc Reimann. A variable neighborhood search for the multi depot vehicle routing problem with time windows. *Journal of heuristics*, 10(6):613–627, 2004.

[42] Jean-Yves Potvin and Jean-Marc Rousseau. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. *European Journal of Operational Research*, 66(3):331–340, 1993.

[43] Sushil J Louis Gregory JE Rawlins. Syntactic analysis of convergence in genetic algorithms. *Foundations of Genetic Algorithms 1993 (FOGA 2)*, 2:141, 2014.

[44] Jacques Renaud, Gilbert Laporte, and Fayez F Boctor. A tabu search heuristic for the multi-depot vehicle routing problem. *Computers & Operations Research*, 23(3):229–235, 1996.

[45] Panagiotis P Repoussis, Christos D Tarantilis, and George Ioannou. The open vehicle routing problem with time windows. *Journal of the Operational Research Society*, 58(3):355–367, 2007.

[46] Yves Rochat and Éric D Taillard. Probabilistic diversification and intensification in local search for vehicle routing. *Journal of heuristics*, 1(1):147–167, 1995.

[47] Robert A Russell. Hybrid heuristics for the vehicle routing problem with time windows. *Transportation science*, 29(2):156–166, 1995.

[48] Surya Sahoo, Seongbae Kim, Byung-In Kim, Bob Kraas, and Alexander Popov Jr. Routing optimization for waste management. *Interfaces*, 35(1):24–36, 2005.

[49] Michael Schmidt and Hod Lipson. Age-fitness pareto optimization. In *Genetic Programming Theory and Practice VIII*, pages 129–146. Springer, 2011.

[50] Karel Slanỳ. Comparison of cgp and age-layered cgp performance in image operator evolution. In *European Conference on Genetic Programming*, pages 351–361. Springer, 2009.

[51] Kay Chen Tan, YH Chew, and Loo Hay Lee. A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems. *European Journal of Operational Research*, 172(3):855–885, 2006.

[52] Kay Chen Tan, Loo Hay Lee, QL Zhu, and Ke Ou. Heuristic methods for vehicle routing problem with time windows. *Artificial intelligence in Engineering*, 15(3):281–295, 2001.

[53] Kit-Sang Tang, Kim-Fung Man, Sam Kwong, and Qun He. Genetic algorithms and their applications. *IEEE signal processing magazine*, 13(6):22–37, 1996.

[54] Sam R Thangiah. *Vehicle routing with time windows using genetic algorithms*. Application handbook of genetic algorithms: new frontiers, 1993.

[55] Sam R Thangiah. A hybrid genetic algorithms, simulated annealing and tabu search heuristic for vehicle routing problems with time windows. *Practical handbook of genetic algorithms*, 3:347–381, 1999.

[56] Sam R Thangiah, Ibrahim H Osman, and Tong Sun. 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*, 69, 1994.

[57] Thibaut Vidal, Teodor Gabriel Crainic, Michel Gendreau, and Christian Prins. A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time-windows. *Computers & operations research*, 40(1):475–489, 2013.

Appendix A

Additional Experimental Analysis

A.1 Determining the parameters for ALPSGA

In order to determine the best parameters for the ALPSGA, which is given in chapter 5, we have performed additional experiments and the result is given in table A.1 and A.2. Table A.1 provides the results obtained after using layers size of 4,5, and 6 using polynomial schemes with the age gap of 50. Table A.2 gives the results of different age gap settings. All these experiments tested with instance pr01. Figures A.1-8 showing the fitness plot for different parameters. From the experiments results and plots we observed that if the age gap is too low, it allows randomness frequently, but very short time to breed within the layers. If it set to too high, it delays the upper layer activation as a result inter layer breeding get delayed.

Table A.1: Comparing three different layers size with an age gap of 50

	Layer 4	layer 5	Layer 6
Best	1195.4[8]	1182.00[8]	1182.00[8]
Average	1241.83[8.2]	1222.83[8]	1222.83[8]

Table A.2: Three different age gap for the polynomial ageing scheme with layer size 5

	$Polynomial_{10}$	$Polynomial_{30}$	$Polynomial_{50}$	$Polynomial_{80}$
Best	1283.5[8]	1211.78[8]	1182.00[8]	1185.41[8]
Average	1303.67[8.6]	1234.45[8.3]	1222.83[8]	1216.14[8]

Instances	ALPSGA-MDVRPTW	ALPSGA-MDVRPTWSD	Difference in vehicle	Difference in distance
pr01	1086.82 8 1193.48 5	**1083.52 8**	0	-5.37
pr02	1871.23 11 1886.96 10	**1868.3 11**	0	-3.64
pr03	2572.25 14 2625.30 13	**2468.84 14**	0	-4.41
pr04	3101.4 19	3101.4 19	0	0
pr05	3483.4 22	3483.4 22	0	0
pr06	4232.4 26	4232.4 26	0	0
pr07	1531.12 9 1491.17 11	**1466 11**	0	-5.2
pr08	2263.8 15	2263.8 15	0	0
pr09	3238.8 20 3041.35 20	3238.8 20	0	0
pr10	1609.54 9	1609.54 9	0	0
pr11	950.67 5	943.47 5	0	-7.2
pr12	1609.54 9	1609.54 9	0	0
pr13	2181.91 12	2182.3 12	0	+0.39
pr14	2483.3 15	2483.3 15	0	0
pr15	2685.74 21	2685.74 21	0	0
pr17	1235.7 7 1256.56 8	1235.7 7	0	0
pr18	1879.76 12	**1858.21 12**	0	-21.55
pr19	2549.51 17	2549.51 17	0	0
pr20	3496.5 25 *	3496.5 25 *	0	0
Best solution		5/9	9/9	

Table A.3: Comparison of MDVRPTW and MDVRPTW with share depot using Pareto ranking

Instances	ALPSGA-MDVRPTW	ALPSGA-MDVRPTWSD	Difference in distance	Difference in vehicle
pr01	1182.00 5	1182.00 5	0	0
pr02	1900.88 10	1900.88 10	0	0
pr03	2692.2 13	2692.2 13	0	0
pr04	3201.1 18	3201.1 18	0	0
pr05	3512.83 21	**3489.61 20**	-23.22	-1
pr06	4190.4 25	1182.00 5	0	0
pr07	1572.45 8	**1563.78 8**	-8.67	0
pr08	2329.4 14	2329.4 14	0	0
pr09	3201.4 18	3201.4 18	0	0
pr10	4072.3 26	4072.3 26	0	0
pr11	943.21 5	**984.87 4**	+41.66	-1
pr12	1612.5 8	1612.5 8	0	0
pr13	2168.34 12	**2002.12 12**	-166.22	0
pr14	2449.15 15	2449.15 15	0	0
pr15	2588.34 20	2588.34 20	0	0
pr16	3345.11 23	3345.11 23	0	0
pr17	1217.47 7	**1236.24 6**	+18.77	-1
pr18	1817.3 13	**1789.27 12**	-28.03	-1
pr19	2570.2 16	2570.2 16	0	0
pr20	3298.83 25*	3298.83 25*	0	0
Best solution		6/20	9/20	11/20

Table A.4: Comparison of MDVRPTW and MDVRPTW with share depot using weighted sum

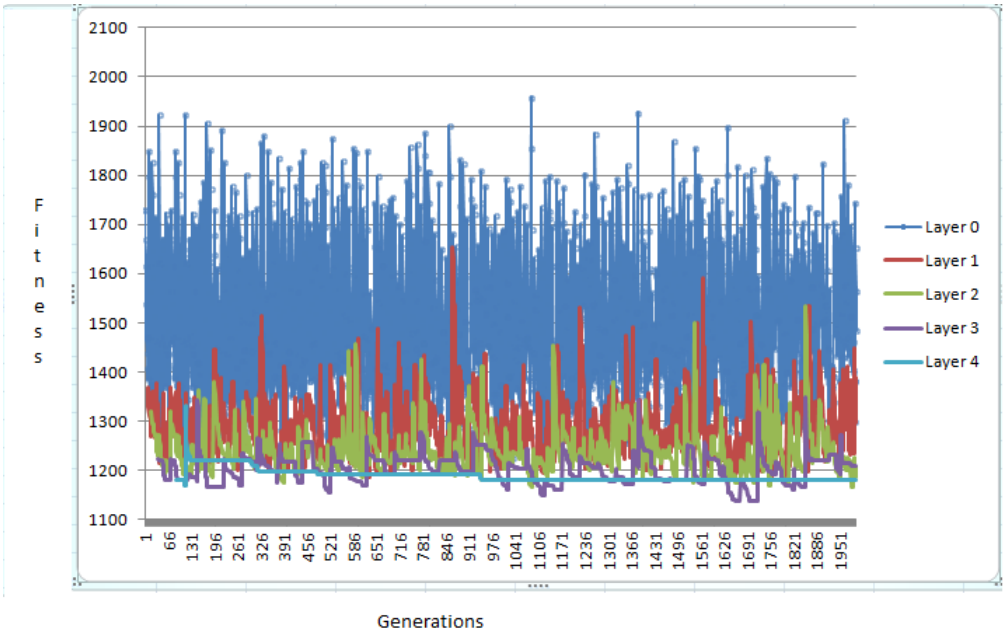

Figure A.1: Fitness plot for pr01 using an age gap of 10 and polynomial aging scheme

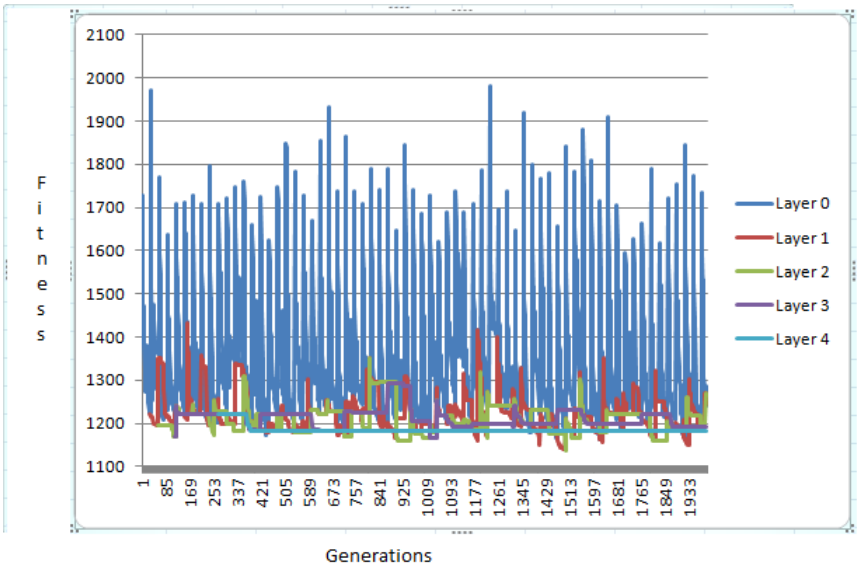

Figure A.2: Fitness plot for pr01 using an age gap of 30 and polynomial aging scheme

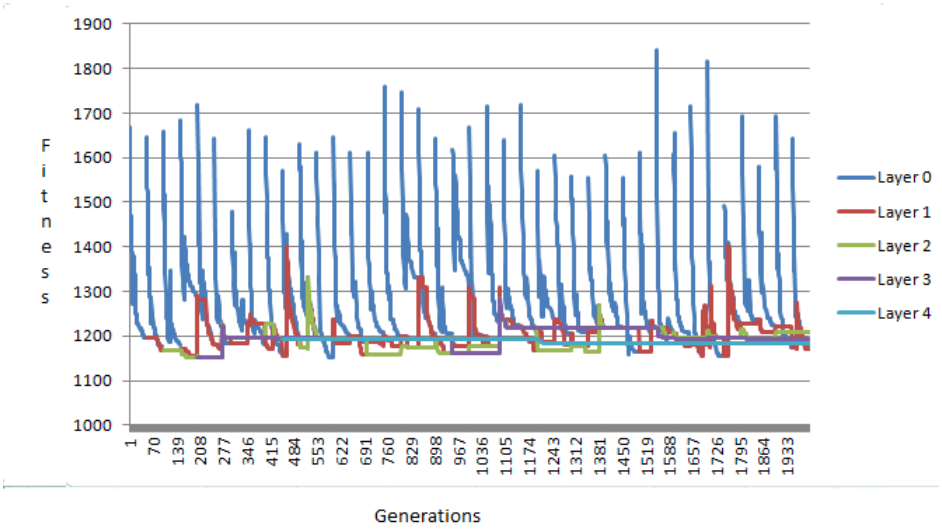

Figure A.3: Fitness plot for pr01 using an age gap of 50 and polynomial aging scheme

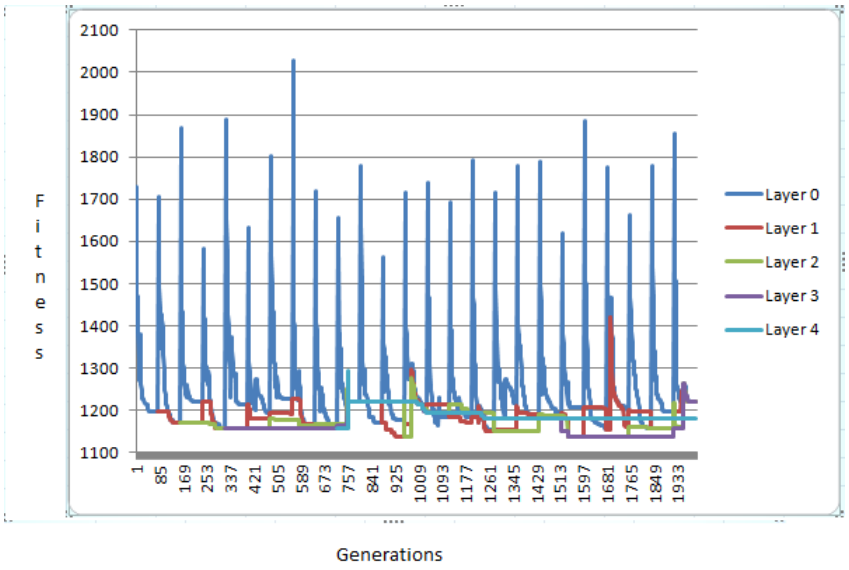

Figure A.4: Fitness plot for pr01 using an age gap of 80 and polynomial aging scheme

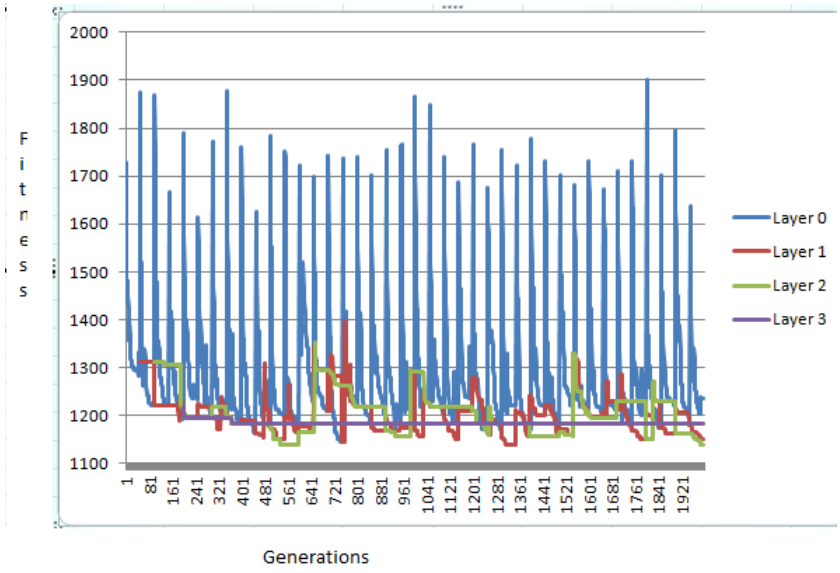

Figure A.5: Fitness plot for pr01 using 4 layers and polynomial aging scheme

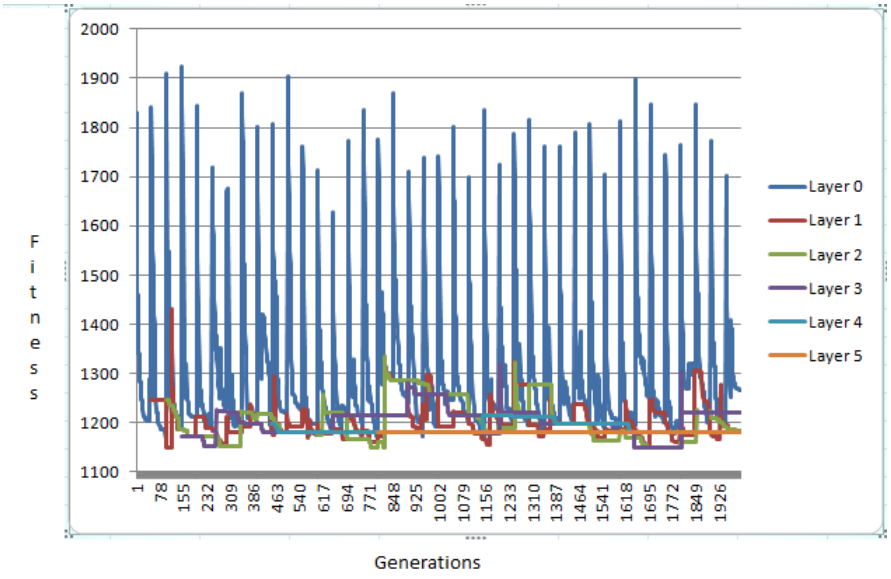

Figure A.6: Fitness plot for pr01 using 6 layers and polynomial aging scheme

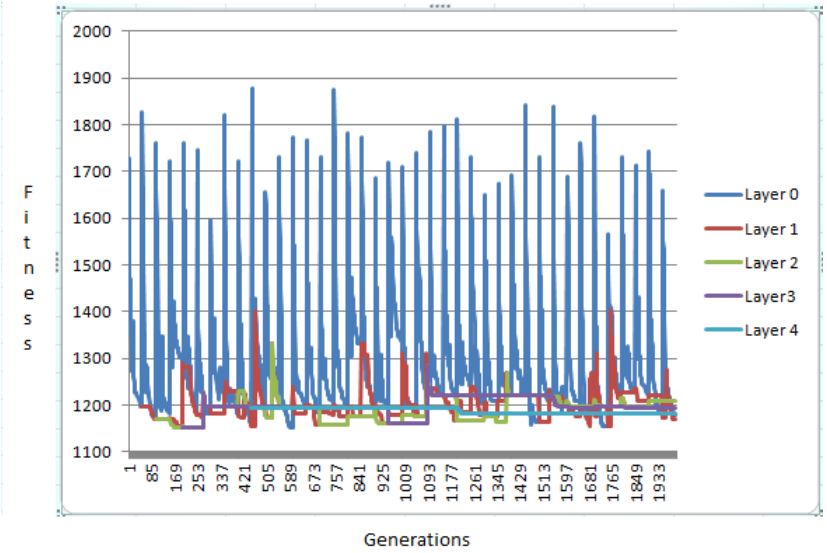

Figure A.7: Fitness plot for pr01 using 5 layers and Fibonacci aging scheme

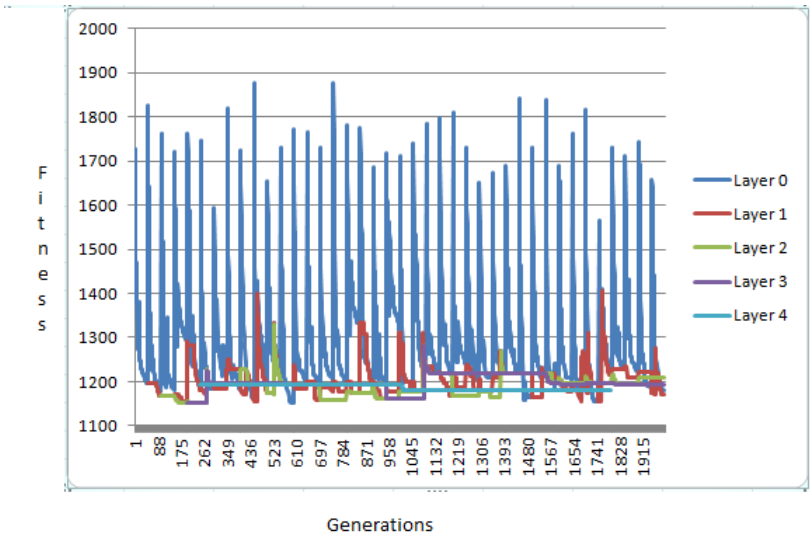

Figure A.8: Fitness plot for pr01 using 5 layers and Linear aging scheme