DEVELOPMENT OF EFFICIENT APPROACHES FOR SOLVING THE VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Technology

in

Computer Science and Applications

Submitted By

Divya Aggarwal

Roll No. 601534004

Under the supervision of:

Dr. Vijay Kumar

Assistant Professor, CSE Department

Mr. Ashish Girdhar

Lecturer, CSE Department

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

June 2017

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "*Development of Efficient approaches for solving the Vehicle Routing Problem with time windows*", in partial fulfillment of the requirements for the award of degree of Master of Technology in *Computer Science and Applications* submitted in Computer Science and Engineering Department of Thapar University, Patiala, is an authentic record of my own work carried out under the supervision of *Dr. Vijay Kumar and Mr. Ashish Girdhar* and refers other researcher's work which are duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

(Divya Aggarwal)

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

(Dr. Vijay Kumar)

Assistant Professor,
CSE Department

(Mr. Ashish Girdhar)

Lecturer,
CSE Department

ACKNOWLEDGEMENT

First of all I would like to thank the Almighty, who has always guided me to work on the right path of the life.

This work would not have been possible without the encouragement and able guidance of my supervisors **Dr. Vijay Kumar** and **Mr. Ashish Girdhar**. I thank my supervisor for their time, patience, discussions and valuable comments. Their enthusiasm and optimism made this experience both rewarding and enjoyable.

I am equally grateful to **Dr. Maninder Singh**, Associate Professor and Head, Computer Science & Engineering Department, a nice person, an excellent teacher and a well-credited researcher, who always encouraged me to keep going with work and always advised me with his invaluable suggestions.

I will be failing in my duty if I don't express my gratitude to **Dr. S.S. Bhatia,** Senior Professor and Dean of Academic Affairs, Thapar University, for making provisions of infrastructure such as library facilities, computer labs equipped with net facilities, immensely useful for the learners to equip themselves with the latest in the field.

I am also thankful to the entire faculty and staff members of Computer Science and Engineering Department for their direct-indirect help, cooperation, love and affection, which made my stay at Thapar University memorable.

Last but not least, I would like to thank my family whom I dearly miss and without whose blessings none of this would have been possible. To my parents, I own thanks for their wonderful love and encouragement. I would also like to thank my brother, since he insisted that I should do so. I would also like to thank my close friends for their constant support.

Date: 30 June, 2016
Place: Thapar University, Patiala

(Divya Aggarwal)

ABSTRACT

Being a key element in logistics distribution, Vehicle Routing Problem becomes an importance research topic in management and computation science. Vehicle Routing Problem with time windows is a specialization of Vehicle Routing Problem. The VRPTW is a fundamental problem underlying many operational challenges in the field of logistics and supply chain management.

Most of the researches have generally focused only on the minimization of total transportation cost. Here, the main focus is the bi-objective optimization considering the minimization of the total transportation cost and number of vehicles. The problem is Non Polynomial time (NP) hard. The bi- objective VRPTW is modelled into mathematical set of equations on which optimisation strategies could work upon. First, a Mixed Integer Programming approach, a general methodology for solving a broad class of VRPs, is developed and use this technique to obtain competent results. This technique could work on limited number of customers. Then, a solution strategy based on Lagrangian relaxation using sub gradient approach is proposed. This approach gives optimal solutions for the considered problem. But, this solution is not efficient to compute solutions for most of the instances with 100 customers. Hence, exact approaches are not much efficient when large instances are considered.

Therefore, Firefly Algorithm, a metaheuristic technique, which could give approximate results to problem is applied to the VRPTW. A modification is proposed to generate optimum results. A strategy of reducing the population space in each consecutive iteration to reduce the overall computation time is applied. Moreover, a distance method, analysed to give best results among four different distance method is used. The proposed approach has been tested on the well-known Solomon's benchmark test instances. Extensive results have shown that the proposed approach outperforms the benchmark results and is comparable to optimum results.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

VRP	Vehicle Routing Problem
MTSP	Multiple Travelling Salesman Problem
DVRP	Dynamic Vehicle Routing Problem
CVRP	Capacitated Vehicle Routing Problem
SVRP	Static Vehicle Routing Problem
MDVRP	Multi-Depot Vehicle Routing Problem
VRPTW	Vehicle Routing Problem with time windows
VRPPD	Vehicle Routing Problem with pickup and Delivery
VRPB	Vehicle Routing Problem with Backhauling
MA	Memetic Algorithm
MIP	Mixed Integer Programming
MOSA	Multi-Objective Simulated Annealing
PSO	Particle Swarm Optimization
GA	Genetic Algorithm
FA	Firefly Algorithm
NP	Non-polynomial
LB	Lower Bound
UB	Upper Bound

Chapter 1
INTRODUCTION

1.1 Vehicle Routing Problem

An important trait of supply chain management is the synchronisation of logistic operations for the transport of commodities within the supply chain. The assignment of designing delivery or pickup service in the area of transportation and supply chain is termed as Vehicle Routing Problem (VRP) in literature. VRP was first stated in the work of Dantzig and Ramser [1] titled "Truck Dispatching Problem" which aims at designing optimal routes for gasoline delivery trucks providing service between single distribution terminal and number of service stations. Collection of household garbage, package delivery, delivery trucks are the main sources of distribution or collection in real world. VRP has played a vital role in distributions and logistics. VRP is the most studied problem in optimization literature because of its practical applicability and its considerable difficulty.

The simple routing problem is known as Travelling Salesman Problem (TSP). It suggests that a salesman visiting number of cities should visit all the cities travelling the minimum distance and returning to the same city where started. The VRP is the extension of multiple traveling salesman problem (MTSP) in which each request is associated with each city and each vehicle has a specific capacity limit. Vehicle Routing Problem refers to the situation where distinct routes are determined for each vehicle starting and ending their journey at one or more depots to service different consumers delivering goods and services. VRP comes under combinatorial problem [2]. The objective of VRP is to serve customers with known demands with minimum cost (or distance travelled) while satisfying some requirements or constraints. In case of adequate supply of vehicles, it also aims to use a minimum number of vehicles. Figure 1.1 shows the illustration of classical VRP having ten customers and three vehicles. Each vehicle travels through different customers. The traversed routes are discriminated through colour lines. The routes are designed in such a way that the total cost/ distance travelled can be minimized.

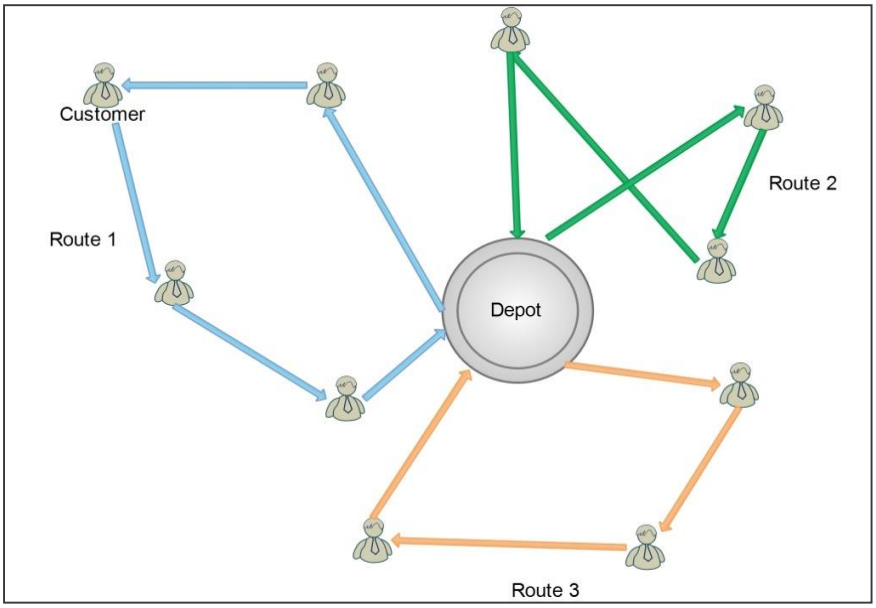


Figure 1.1. Illustration of Vehicle Routing Problem

1.1.1. Mathematical Formulation

In the classical VRP, the customer sites are said to be known as the travel distance (or cost) between the customer locations are known in advance. Moreover, the time required to provide service to each customer is also known. The objective is to generate a set of routes, one for each vehicle starting and ending their journey at depot while ensuring each customer should be considered once by any of the vehicles.

Constants

c_{ij} Cost/Distance from location i to location j

K Total number of vehicles

N Total number of customer locations

a_i Volume of shipment to customer i

b_k Capacity of vehicle k

Decision Variables

$$x_{ij}^k = \begin{cases} 1, & \text{if vehicle } k \text{ travels from location } i \text{ to } j \\ 0, & \text{otherwise} \end{cases}$$

$$y_{ik} = \begin{cases} 1, & \text{if location } i \text{ is served by vehicle } k \\ 0, & \text{otherwise} \end{cases}$$

The mathematical model of VRP [2] is presented as follows:

$$\text{Minimize } \sum_{k=1}^{N} c_{ij} x_{ij}^{k} \tag{1.1}$$

Subject to

$$\sum_{i} a_i y_{ik} \le b_k, \ k = 1,..., K \tag{1.2}$$

$$\sum_{k} y_{ik} = \begin{cases} K, & i = 0 \\ 1, & i = 1,...N \end{cases} \tag{1.3}$$

$$y_{ik} \in \{0,1\}, \ \ i = 1,..., N; \ k = 1,..., K \tag{1.4}$$

$$\sum_{i} x_{ij}^{k} = y_{jk}, \ \ j = 1,.., N; k = 1,..., K \tag{1.5}$$

$$\sum_{j} x_{ij}^{k} = y_{ik}, \ i = 1,..., N; k = 1,..., K \tag{1.6}$$

$$\sum_{ij \in S \times S} x_{ij}^{k} \le |S| - 1, \ S \subseteq \{1,.., N\}; 2 \le |S| \le N - 1; k = 1,..., K \tag{1.7}$$

$$x_{ijk} \in \{0,1\}, \ \ i = 0,..., N; j = 0,...., N; k = 1,..., K \tag{1.8}$$

The objective function focuses on minimizing the incurred cost. Constraint (1.2) ensures that quantity to be delivered by vehicle to every customer on each route should not go over the vehicle capacity. Constraints (1.3) and (1.4) make sure that every vehicle starts and ends its journey at the depot, and also that each customer location should be visited exactly once by any one of the vehicle. Constraints (1.5)-(1.7) satisfy for the travelling salesman problem scenario for a particular vehicle k at a time.

1.1.2. Variants of VRP

Inclusion of the real life scenarios to the VRP has resulted in different variants which have been evolved from the classical VRP. They are classified based on the network (between the depot and customer nodes) structure, the fleet composition and structure, nature of transported goods and the characteristics of customers. Figure 1.2 shows the classification of Vehicle Routing Problem on the basis of availability of problem components and vehicle capacity.

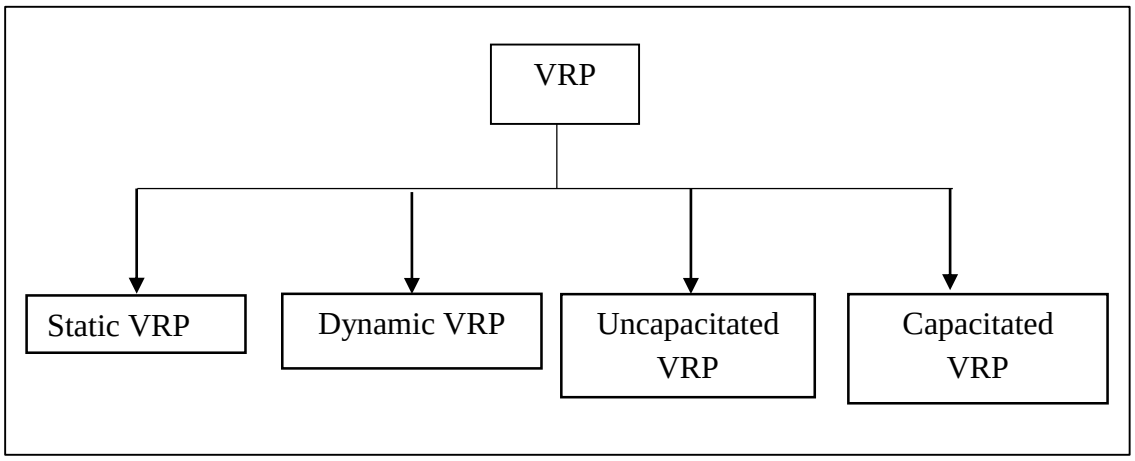


Figure 1.2. Classification of Vehicle Routing Problem based on availability of problem components and vehicle capacity

The VRP is broadly classified into the categories. These are described below

Static and Dynamic VRP

Vehicle Routing Problem can be classified into Static and Dynamic VRP on the basis of availability of problem components. In Static VRP, problem components are known in advance. So, planned routes are easily possible for all the values known. Here, new orders will not arrive once the serving process starts. In static vehicle routing problem, n customers get served by a single depot. Here, each customer i has a demand q_i which is to be satisfied by one vehicle. A group of m vehicles, each vehicle j with capacity Q_j, deliver goods to the customer locations. A service time S_i is related to each customer i to represent the time taken to service that customer. Therefore, a solution to VRP consists of a set of tours. Static VRP is deterministic in nature.

In Dynamic Vehicle Routing Problem (DVRP), any partial or full relevant

information related to the system conditions is made available in real time only when it is needed. Some of the information can be possible known in advance. The Dynamic VRP is directly related to the static VRP. The main difference is that in dynamic VRP, new orders keep on arriving after computation has already started. Thus, The DVRP can be stated as a series of static VRP-like instances. Each static VRP contains the customers known but not served. Here, the term time slice limits the time devoted to each static problem component. The best solution is examined at the end of each time slice and the orders arrived during the time slice is submitted to next static instance. Dynamic VRP is also called On-line VRP [3]. There are two classes of dynamic problem. First presents the time dependency of travel duration [4] which suggests various reasons to affect vehicle speed along roads and second considers the availability of vehicles as dynamic component [5].

Uncapacitated and Capacitated VRP

On the basis of vehicle capacity, Vehicle Routing Problem can be classified into Uncapacitated and Capacitated VRP. In Uncapacitated VRP, problem formulation comprises of solving minimum cost route for each of the vehicles such that every customer node is visited only once and all the customer sites should be visited. The objective is to minimize the total cost/distance travelled by vehicles. Uncapacitated VRP considers two problem: vehicle allocation and routing. The vehicle allocation problem decides the allocation of vehicle to the customer whereas routing problem suggests the order to serve. Both the problem aims to travel minimum total distance. Uncapacitated VRP consists of solving TSP for each vehicle i.e. solving minimum cost route for each vehicle.

In the Capacitated VRP (CVRP), a given number of similar vehicles with uniform capacity would provide service to the customers with given demands at a minimum cost (or distance). In CVRP, the vehicles are given with a capacity constraint, so it limits vehicles to carry the commodity less than the capacity. In CVRP, the transportation request consists of goods distribution from a single depot to the n customers. The fleet is homogeneous with same capacity and identical operating costs. Each vehicle will serve customer subset $S \subseteq n$ starting and ending at the depot while satisfying the commodity demands at each location in S. A Vehicle moving from customer location i to j incurs cost c_{ij}.

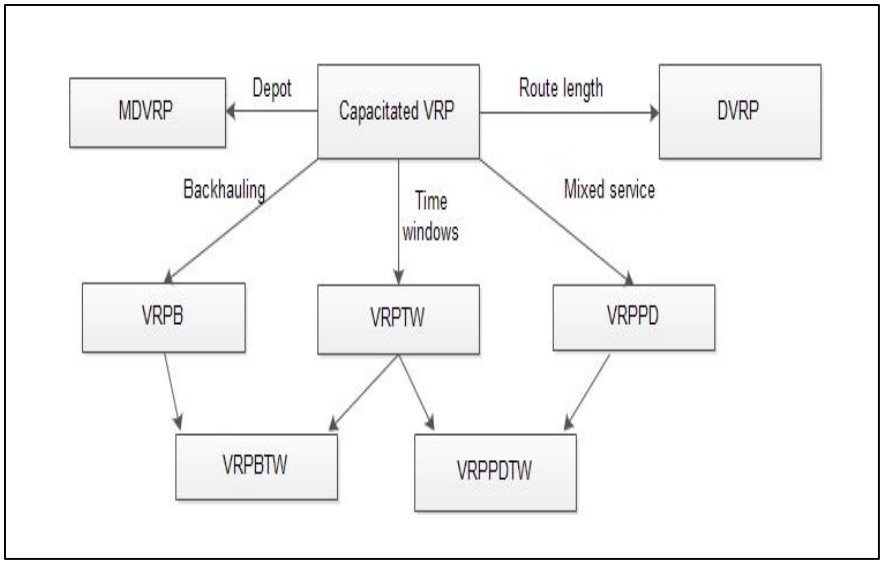


Figure 1.3. Other Constraints added to Vehicle Routing Problem

In addition to the above variants, other constraints are to be considered that can lead to various variants which are shown in Figure 1.3. Some of the variants are given below:

i. **Vehicle Routing Problem with time windows (VRPTW)**: The VRP with time windows (VRPTW) [6, 7] is just like VRP with additional constraint where every customer has particular time window to schedule his service. The objective of VRPTW is to serve the customers within given time interval while minimizing the cost (or distance travelled), besides following capacity or total journey time constraints. Thus, this problem is non-polynomial-hard (NP-hard) problem. Therefore, heuristics should be used to give best solutions. The main focus of this dissertation is on VRPRTW.

ii. **Vehicle Routing Problem with Pickup and Delivery:** VRP with pickup and delivery adds the constraint of returning the goods which should fit into vehicle capacity, so it often lead to poor utilization of vehicle capacity. Therefore, a restriction should be made of delivering goods when travelling from depot to customers and picking up returns when travelling back from customer nodes to depot.

iii. **Vehicle Routing Problem with Backhaul:** VRP with backhaul is a VRP in which customer can both demand and return the goods. It is similar to the VRP with

pickup and delivery but with a restriction that all the deliveries must be completed for a particular route before any return can be made.

iv. **Multi-Depot Vehicle Routing Problem:** This variant of VRP includes the possibility of more than one depots. If the customer nodes are clustered around depots, then the distribution problem can be structured as set of independent VRPs. But if customer and depot nodes are intermixed then existence of Multi depot VRP comes into being.

v. **Vehicle Routing Problem with multiple trips:** Here**,** the vehicle can traverse back and forth through the depot. This variant of the VRP enables the vehicle to traverse more than one route.

vi. **Stochastic Vehicle Routing Problem:** In Stochastic VRP, some of the problem variables are uncertain and described as random variables. The most common variants of Stochastic VRP are VRP with stochastic customer demands, VRP with stochastic customers, VRP with stochastic customer and demand, and stochastic service time.

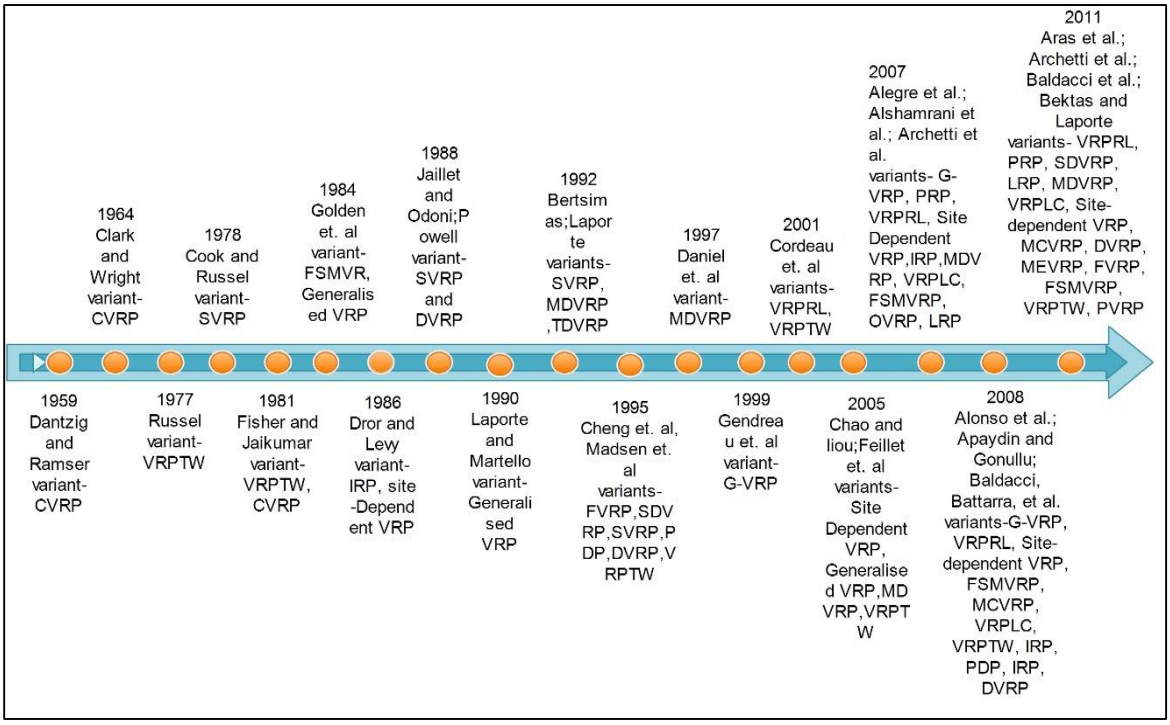


Figure 1.4. Evolution of Vehicle Routing Problem and its variants

Some of other variants include constraint like single vehicle and limited capacity, route duration limit. Figure 1.4. shows the yearly evolution of various VRP variants.

1.2. Applications of VRP

Some of the applications of the VRP is given below:

i. **Distribution Service:** Collection of household waste, package delivery, gasoline delivery trucks are the primary utilisations of VRP in daily life.

ii. **Pickup and Delivery service:** VRP with pickup and delivery has numerous applications, for example, the delivery of drinks and simultaneous collection of empty bottles and the bus service between the local airport and several hotels. Here also the capacity constraint ensures no overloading.

iii. **One-to-many-to-one:** 1-M-1 problems are characterised by the system where commodities are delivered from one depot to many customer and simultaneously collecting commodities from the customers and then transporting back to the depot. Typical applications of these problems are less than truckload applications and urban courier operations. More uses of VRP include the transport directing for understudies, patients, handicapped person and elderly.

iv. **2-Elchon VRP:** 2E-VRP has been applied in city logistics where the delivery from the single station to various locations is handled by intermediate depots also referred as satellites.

v. **Disaster Management:** In the situation of disaster a fixed number of vehicles needs to reach cities or affected areas at right time as could be expected under the circumstances.

vi. **Dynamic Entity:** In last year's programming devices coordinating telematic administrations have been created with the point of empowering a speedier response of planners to the flow of the transportation framework and to the possible interruptions brought about by the failure of vehicles or by the intense traffic circumstances on the roads. Web based planning tools are crucial for real time applications, for example, the control of automatic guided vehicles.

1.3. Motivation

VRP plays a vital role in distribution and logistics. VRP is most studied problem in optimisation literature because of its practical applicability and its considerable difficulty. Toth and Vigo reported in their work in 2002 that use of computerised methods would result in decrease of transportation cost by 5 to 20% [8]. Therefore, there is a need of optimization in the system to manage the delivery task. Here, the main concern is the more specific variant of Vehicle Routing Problem, i.e., Vehicle Routing Problem with time windows (VRPTW).

1.4. Dissertation layout

The organisation of the dissertation is described as follows:

- Chapter 2 gives a brief study of the earlier work done in the field of Vehicle Routing Problem and its variants. It also records the work referred to propose the Problem and work that helps in the analysis of the problem.
- Chapter 3 formulated the problem statement. Based on the gaps identified in the literature for the Vehicle Routing Problem, a descriptive problem statement has been proposed to effectively work on each identified gap.
- Chapter 4 gives the various proposed work in the light of this research work. Here, the complete description of various solution approaches used to solve the proposed problem statement is given.
- Chapter 5 gives the results obtained from the proposed work. Performance analysis and evaluation of results is also described. This sections also deals with the dataset, tools and technology used in finding the solution for the problem.
- Chapter 6 concludes the work and also discusses the future scope of the work. This section describes the contribution done in the respective field.

Chapter 2
Literature Review

The Vehicle Routing Problem dates back to 1959 when Dantzig and Ramser [1] introduced the term, for solving the routing problem of gasoline delivery trucks to a number of service stations. Since then, the interest group in VRP has evolved from the small group of mathematicians to broad range of researchers. The problem has been widely studied in [9, 10]. Many solution methods have been proposed for finding optimal solutions for the VRP or its various variants. The solution approaches for the VRP can be subcategorized as exact methods (branch-and- bound, branch-and-cut), heuristics and Meta heuristics. Like the other combinatorial optimization problems, exact solutions to VRP are also intensive to compute [11]. Therefore, to attain the efficiency, one must choose approximation and heuristics methods which work well in practice. Figure 2.1 shows the complete classification of the solution approaches.

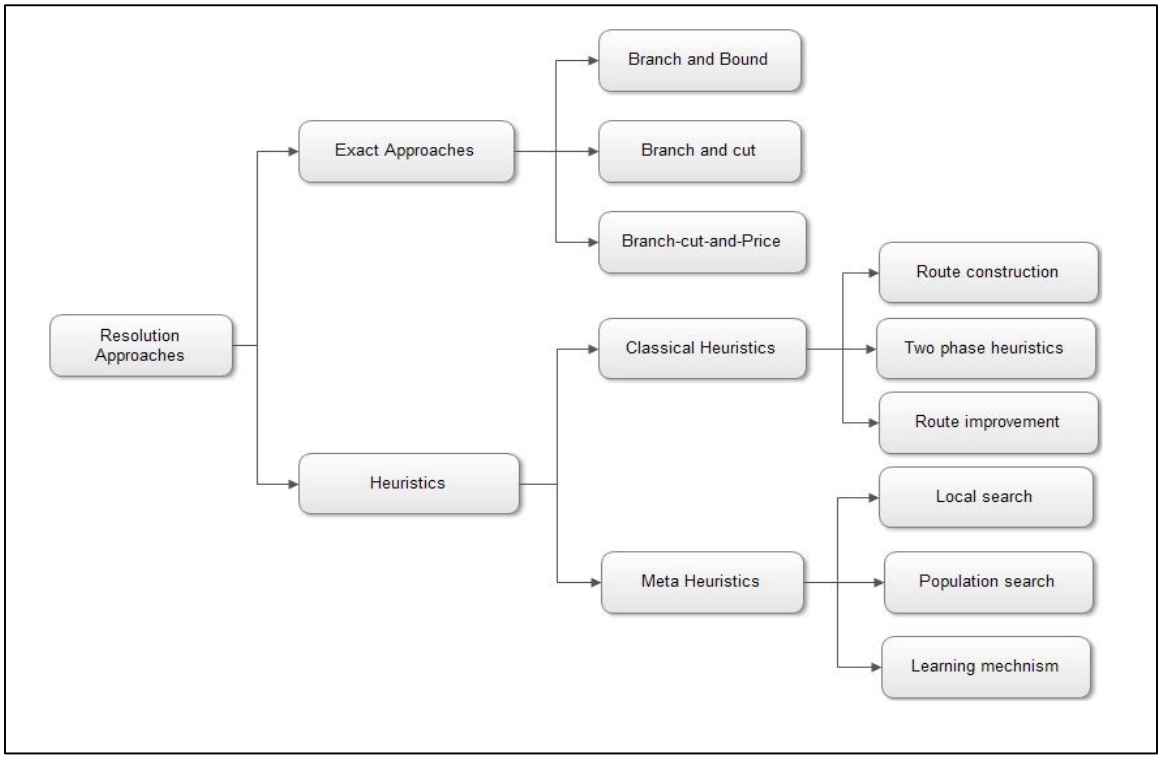


Figure 2.1. Classification of various Resolution approaches

Here, the main area of concern is Vehicle Routing Problem with time windows due to its practical applicability and specific nature. VRPTW is considered NP-hard because of the NP-hardness of VRP. The VRPTW is the expansion of the VRP with addition of time windows to provide the service at each customer within a specific time interval. Various researches have been done in the field of VRPTW which can be used to specify many real-world problems. The early works on the VRPTW include a variety of case studies [12-14]. Earlier studies on VRPTW include both optimization algorithms and heuristic approaches. However, the focus of current research is heuristic and Metaheuristic approaches. Various surveys have been done in the field of VRPTW [15-17].

2.1. Review on VRP using Exact methods

VRPTW has been solved using heuristic and exact optimization approaches. Due to the complexity of the VRP family problems, exact methods become useless especially when large-scale instances are tackled. Kolen *et al.* [18] used branch and bound approach to solve a VRPTW having customer nodes varying between 6 and 15. When the number of nodes is set to 6, then computation took about one minute. The branch and bound approach was unable to process for 12 customer nodes. Sam [19] worked on reducing the delivery costs by considering only travel time taken by vehicle while traversing from one customer to another customer. Moreover, this work involves product delivery by violating the time windows and compensating it by adding a penalty in the objective function. Dondo *et al.* [20] proposed a Mixed Integer Linear Programming (MILP) for solving multi-depot heterogeneous-fleet VRPTW problem. Dondo and Cerda [21] offered three-phase heuristics for *m*-depot routing problem with time windows and heterogeneous vehicles. The clustering algorithm was applied on VRPTW which could solve at most 25 nodes by restructuring MILP.

Cetinkaya *et al.* [22] proposed a hybrid approach that combines both Mixed Integer Programming and Memetic Algorithm (MA) for solving two-stage Vehicle Routing Problem with Arc Time Windows. He added time variant to each and every arcs. Suwansuksamran *et al.* [23] presented a mixed integer programming for a case study company. In this problem, a binary variable is used to indicate whether a customer has time window constraint or not. Simbolon and Mawengkang [24] suggested a mixed integer

approach for vehicle routing problem with window having a variant of avoiding routes. When pair of edges occurs frequently, then the sub route is avoided to eliminate the heavy traffic. The experimental results revealed that MIP formulation gave good results for both small and medium-size instances. However, it was concluded that MIP formulation is good enough for instances having up to 50 customers.

Sabar *et al.* [25] suggested an approach that consists of two phases, i.e., math phase and hyper heuristic phase. This approach is better than the simple hyper heuristics approach. Hokama *et al.* [26] suggested branch-and-cut approach for solving vehicle routing problem having loading constraints. It solves the problem in smaller computation time, but packing strategies and lower bounds could be improved. Ozbaygin *et al.* [27] proposed a branch-and-price algorithm for vehicle routing problem. This technique used the concept of roaming delivery locations. The proposed approach provided optimal solution. However, it requires the prior knowledge of travel time and itinerary of customer.

The various effective approaches found in the literature consist of constrained shortest path relaxations [28, 29]. Column generation [28] and Lagrangian decomposition [29] are the similar methods. In both the methods the complete problem is partitioned into two sub problems- the master problem and the sub problem. The sub problem is the shortest path problem satisfying the time and capacity constraints. There are various papers with little variation in approaches for Lagrange relaxation such as variable splitting accompanied by Lagrange relaxation, k-tree procedure along with Lagrangian relaxation [11]. Kohl and Madsen [30] suggested an approach solving shortest path with side constraints which is followed by Lagrange relaxation. This involves relaxing the constraint ensuring that every customer should be served exactly once and adding to objective function. Various sophisticated approaches based on Lagrangian Relaxation, which now can solve large instance of problem to optimality have been introduced. Martinhon *et al.* [31] proposed a relax-and-cut algorithm which combines the Lagrangian based approach with comb and multi star inequalities. This approach improves the quality of lagrangian bound. Kallehauge *et al.* [32] proposed a cutting plane algorithm to solve the Lagrangian dual. Cutting planes are generated by solving the sub problem and then they are introduced in a master problem to impose bounds on the dual variables. Imposing bounds will ensure the stability in between the iterations. Karimi and Seifi [33] proposed the acceleration of

lagrangian relaxation for the vehicle routing problem with time windows. Here analytical center cutting plane method is used for the acceleration. The results are then compared with basic cutting plane algorithm applied with column generation which suggests the improved results in terms of computational time and quality of lower bounds.

2.2. Review on VRP using Approximation methods

Besides these, various researchers have applied heuristics techniques for VRPTW. Since 1990s, numerous metaheuristics algorithms have been suggested for better approximate solution to the vehicle routing problem with time windows. Holland [34] proposed a Genetic Algorithm which initializes the random chromosome and generates the solution for VRPTW in bit strings. Thereafter, chromosomes were selected for crossover and mutation process to produce children. The process keeps on repeating until the evolution has been converged. El-Sherbeny [35] solved VRPTW problem using a multi-objective simulated annealing (MOSA) method. Here, three objectives were defined such as minimizing the vehicle used, total time, and concerning the flexible duration of routes. Beatrice *et al.* [36] presented a hybrid approach that comprises of genetic algorithm (GA) and Pareto ranking to solve VRPTW. This method was applicable on problems with more than 100 customer nodes.

Chiang and Russell [37] proposed three different simulated annealing methods, one uses modified version of the k-node interchange mechanism and second that use the k-interchange mechanism proposed by Osman. The third algorithm uses Tabu list from the Tabu search metaheuristic. The second and third method has faster convergence but the first gives better results. Another approach, Tabu search, is among the one of the many oldest metaheuristics. It was first proposed by Glover [38]. In this, neighbourhood of the current solution is evaluated at each iteration and best solution is selected and termed as the new current solution. To eliminate the local optimum, the existing solution is considered to be best even it is worse than current solution. In contrast to the Simulated Annealing and Tabu search, the Genetic algorithms work on the population of solution and not only with one solution [39]. A population of structures, each one corresponding to a possible solution, represents the space of the solution of the problem. The next population is generated by application of genetic operators (selection, crossing and mutation) on the

potential parents chosen inside the population. The genetic algorithms are based on the principle that best parents produce best children. The best parents are crossed to give the place to new children who replace the parents. This procedure is repeated until a population is obtained which is strong enough, corresponding to optimal or almost optimal solution of the problem.

Amini *et al*. [40] suggested a Particle Swarm Optimization approach for a real world case study of a chlorine capsule distribution company. The results significantly increase the profit upto 10%. The results for the algorithm are compared with the benchmark results with 25 customers and show significant results with small error rate to optimal solutions. Zhu *et al.* [41] proposed a collaborative population based approach as it combines the local search methods with global search methods in order to balance exploration and exploitation. In this, the results are discussed for two cases- first compares the particle swarm optimisation with the genetic algorithm and show it generating the better results. Second case considers the random generation of population. It is analysed that the improved Particle swarm approach gives optimal results.

The Firefly was first introduced by Yang (2008) for optimization of multimodal function [42]. Initially firefly algorithm was developed for the continuous optimization. But later it also works on the discrete optimization. Firefly Algorithm is a population based algorithm which is inspired from the behaviour of fireflies. Ever Since the first implementation of the FA, it had been applied in variety of areas. Some of the applications include the continuous optimization [43], multi-modal optimization [44], combinatorial optimization [45], and multi-objective optimization [46].

Initially the Firefly was introduced to work on Continuous problems but now various modified versions of FA are available in the literature to work on discrete optimization problems. In [47], a discrete FA was suggested to solve Quadratic Assignment Problem. Sayady *et al*. [16] proposed other discrete approach for minimization of the makespan for flow shop scheduling problem. This problem is also considered as NP-hard problem. Marichelvam et al. [48] presented another discrete FA for the multi-objective flexible job shop scheduling problem. Moreover, a evolutionary discrete FA for the symmetric TSP was presented in [49]. Various modified approaches and hybrid algorithms have been

suggested in the literature. Tilahun and Ong [50] proposed a modified Firefly Algorithm. In [51, 52], a Chaos randomized firefly algorithm was developed. Additionally, an ant colony hybridized with a FA was suggested by Aruchamy and Vasantha [53]. Then, an algorithm hybridizing FA with neural networks was proposed by Hassanzadeh *et al.* [54]. Yang has also compared FA with the fully-developed particle swarm optimization algorithm (PSO) and genetic algorithm (GA) [55, 56].

Chapter 3
Problem Statement

VRP is the most studied problem in optimization literature because of its practical applicability and its considerable difficulty. The majority of the real world problem are much more complex than the classical VRP. Therefore, constraints, such as vehicle capacity, service time or time interval at each customer, are always augmented to the classical VRP which reveals the Capacitated Vehicle Routing Problem (CVRP) and the Vehicle Routing Problem with Time Windows (VRPTW) variants. In last decades, many real world problems have required extended formulation of the VRP which has resulted in variants like multiple depot VRP, periodic VRP, split delivery VRP, stochastic VRP, and VRP with backhauls, VRP with pickup and delivering and many others.

In literature, various solution approaches can be found for one or the other VRP, where the algorithms are designed to deal with the specific subject or specific constraint. Although above mentioned VRP variants mimic some of the real world situations, but here, the focus is on the specific variant- Vehicle Routing Problem with time windows (VRPTW). The vehicle routing problem with time windows (VRPTW) is the specialization of classical VRP, being very applicable in real world problems. Each customer has a fixed service time interval. Moreover, the hard time window constraint is considered i.e. the time interval cannot be violated. Due to the applicability of VRPTW in real life situations, it has been widely investigated through various researchers.

Gap identified

- In the past, there were two major areas of concern in logistics- high transportation cost and long computation times. Moreover, it has been noticed that there is scope to design VRPTW problem as a bi-objective problem that minimize both the transportation cost and used number of vehicles.
- The identification of route feasibility is an important issue for VRPTW.
- The selection of time window during the design of routes is another research issue.

Solving VRP has been a challenging today's task. A number of different exact and heuristic methods have been studied to solve the VRP that is known to be NP-hard. Here, the mixed integer programming and Lagrangian relaxation using subgradient approach is solved for the VRPTW. Although the exact methods give the optimal solution, their computation time considerably increases with the increasing size of the problem.

Various heuristic methods exist for solving problems that are known to be NP-hard. The literatures have showed Firefly Algorithm as promising algorithm as it has also outperformed Particle Swarm Optimisation. Firefly is an evolutionary algorithm which is based on the behaviour of fireflies. The fireflies has feature, attractiveness associated with them proportional to its brightness and the fireflies gets attracted to the brightest firefly. Here, in computation theory, fireflies are the symbolism of the problem solutions and the brightest firefly gives the best solution. Moreover, a modification to the Classical firefly Algorithm is proposed with the introduction of distance method called the Chebyshev Distance method and reduction of problem space at each iteration. The Chebyshev distance method is analysed to be efficient among the other methods of distance calculation. Reduction of problem space has lead the problem to be more computationally efficient and convergence speed has been decreased.

Chapter 4
Proposed Work

The vehicle routing problem with time windows (VRPTW) is the specialization of classical VRP, being very applicable in real world problems. It involves a fleet of vehicles starting from a depot to serve a number of customers which are diversely located. Each customer has a fixed demand request. Due to the applicability of VRPTW in real life situations, it has been widely investigated through various researchers. Figure 4.1 describes the illustration of VRPTW having hard time window constraint. The routes in VRPTW are discriminated through colour lines. Each customer has fixed time constraint window.

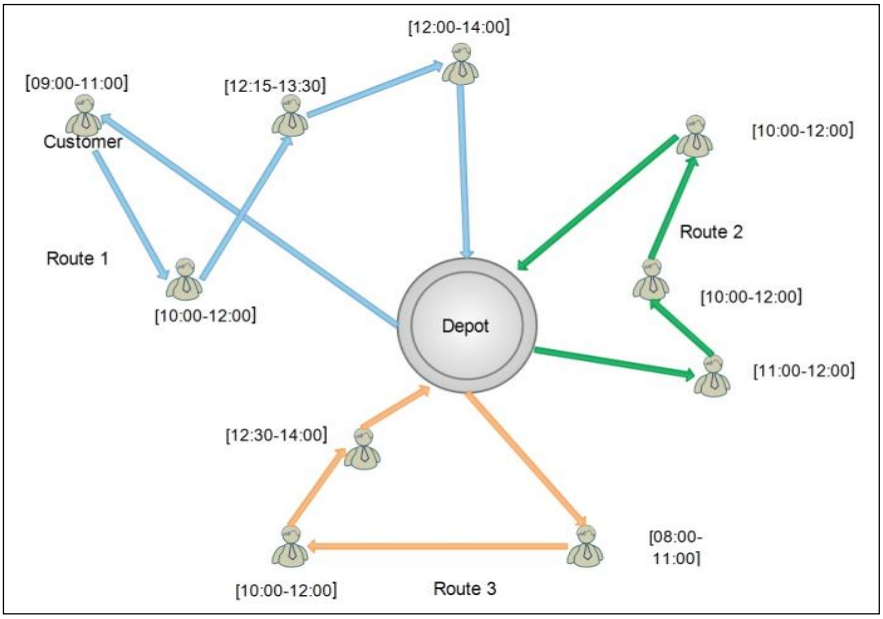


Figure 4.1. A typical vehicle routing problem with time windows

4.1. Model Formulation

The problem is to determine the set of routes that minimize transportation cost and number of vehicles used while satisfying the following constraints:

- The load of a vehicle cannot increase its predefined capacity.

- Vehicle will not service the customer before the earliest time and not later than the latest time.
- Number of Vehicle should be minimized.
- Each vehicle should visit customer only once.

From a graphical point of view, VRPTW is defined as a graph (C', E) which is to be traversed by a fleet of vehicles V. Here, the vehicles are considered homogeneous, i.e., every vehicle has same capacity.

Nomenclature indices

i, j	$(i, j) \in E$
i	*customer* i

Model Parameters

C	Set of all customers $(1,2,...,n)$
C'	Set of all nodes $C' = C \cup 0 \cup n+1$
Q	Vehicle capacity
V	Number of vehicles
c_{ij}	Cost of travel from customer i to customer j
t_{ij}	Travel time from customer i to customer j
r_i	Demand to service customer i
$[e_i, l_i]$	Time window requested by customer i, where e_i indicates the earliest service starting time and l_i indicates the latest service starting time

Decision Variables

The model includes two types of decision variables X and S. The decision variable X_{ijk} is defined as follows:

$$X_{ijk} = \begin{cases} 1, & \text{if edge from customer } i \text{ to customer } j \text{ is traversed by vehicle } k \\ 0, & \text{otherwise} \end{cases}$$

Another decision variable S_{ik} represents the time at which vehicle k service customer i.

Let us assume that $S_{0k} = 0$ and $S_{(n+1)k}$ denote the time at which vehicles arrives at returning depot. The objective of model is to design a set of minimal cost routes, one for each vehicle, such that all customers are serviced exactly once while minimizing the number of vehicles. Therefore, the objective function is considered as bi-objective. The split deliveries are not allowed. The routes must be feasible with respect to the capacity of vehicles and time windows of the customers serviced.

The mathematical model of VRPTW is given below:

$$\textit{Minimize} \qquad \sum_{k \in C'} \sum_{(i,j) \in E} c_{ij} X_{ijk} \tag{4.1}$$

Subject to

$$\sum_{k \in V} \sum_{j \in C'} X_{ijk} = 1, \quad \forall i \in C \tag{4.2}$$

$$\sum_{i \in C} r_i \sum_{j \in C'} X_{ijk} \le Q, \quad \forall k \in V \tag{4.3}$$

$$\sum_{j \in C'} X_{ojk} = 1, \quad \forall k \in V \tag{4.4}$$

$$\sum_{i \in C'} X_{ipk} - \sum_{j \in C'} X_{hjk} = 0, \quad \forall p \in C, \forall k \in V \tag{4.5}$$

$$\sum_{i \in C'} X_{i,n+1,k} = 1, \quad \forall k \in V \tag{4.6}$$

$$X_{ijk} (S_{ik} + t_{ij} - S_{jk}) \le 0, \qquad \forall (i, j) \in E, \forall k \in V \tag{4.7}$$

$$e_i \le S_{ik} \le l_i \ , \qquad \forall i \in C', \forall k \in V \tag{4.8}$$

$$X_{ijk} \in \{0,1\} \qquad \forall (i, j) \in E, \forall k \in V \tag{4.9}$$

The objective function (4.1) states that the total cost should be minimized. Constraint (4.2) suggests that only one vehicle should service one node while constraint (4.3) ensures that

the capacity constraint is fulfilled. Here, no vehicle can serve the customer demands beyond its capacity permits. Constraints (4.4)-(4.6) fulfill the flow constraint for the path traversed by vehicle k. Constraint (4.7) makes sure that vehicle k, $k \in V$ should not arrive at customer j before $S_{ik} + t_{ij}$ if travelling from node i to j. Constraint (4.8) checks for the satisfaction of time constraint. An unused vehicle is represented by an empty route from node 0 to $n+1$.

A novel constraint for minimization of vehicles used is defined below:

$$\sum_{k \in V} \sum_{j \in C'} x_{0jk} \leq |V|, \quad \forall k \in V, \forall j \in C' \tag{4.10}$$

Note that constraint (4.7) may results in non-convex optimization because it involves quadratic terms. These terms are rewritten in linear form which is given below.

$$S_{ik} + t_{ij} - M_{ij}(1 - X_{ijk}) \leq S_{jk} \quad \forall i, j \in C', \forall k \in V \tag{4.11}$$

Here, M_{ij} is described as a large constant that can be replaced by $max(l_i + t_{ij} - e_i, 0)$.

4.1.1. Constraint Workflow

Figure 4.2 shows the constraint workflow of VRPTW. The constraint workflow consists of five main stages. The first stage includes the integrity constraint of given problem. It is satisfied that each customer should be served exactly once by each vehicle. In the second stage, the capacity constraint of vehicle is maintained. It ensures that the vehicle should deliver the goods in their predefined capacity. The third stage ensures that each vehicle should have an edge between an intermediate node and the start node. The fourth stage is responsible for time window constraint. It ensures that no vehicle is allowed to serve before the earliest time and later than the latest time for a particular customer. The fifth stage is responsible for minimizing the vehicle used.

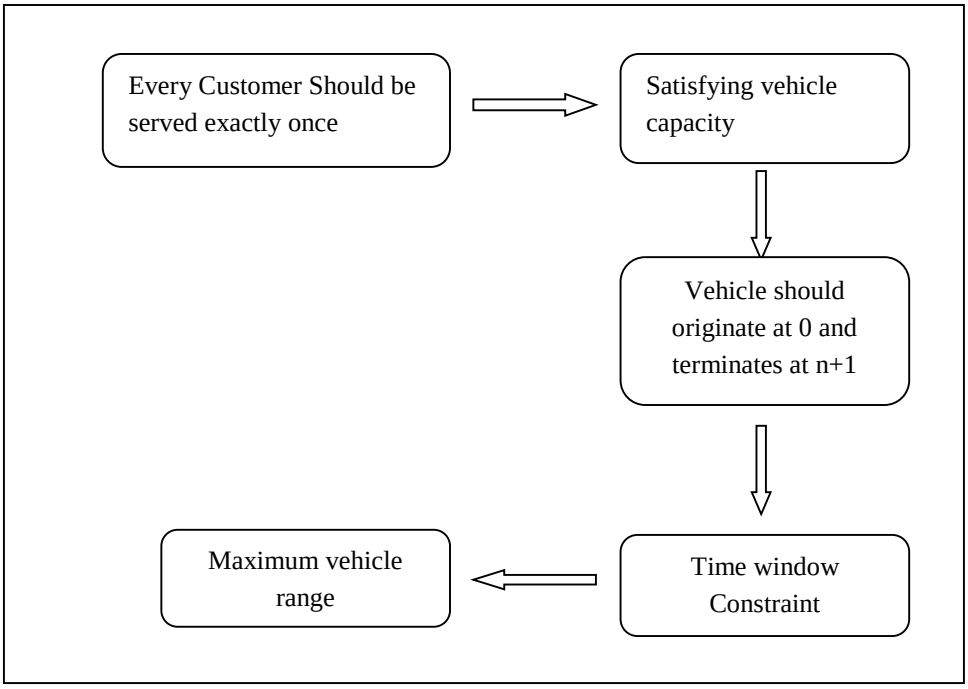


Figure 4.2. Constraint workflow of Vehicle Routing Problem with time windows

4.2. VRPTW using Mixed Integer Programming

The novel mathematical formulation of mixed integer programming proposed to satisfy service time constraint is described. Here, the main aim of proposed approach is to minimize the total transportation cost and number of vehicles used.

4.2.1. Algorithm

Step 1. a) Obtain the initial linear relaxation of the mixed integer programming formulation.

b) Construct the initial feasible solution and designate this solution as upper bound.

c) Add the root node to tree and set the root node as current node.

Step 2. *If* termination criterion is satisfied, *then*

Return the upper bound and Exit

Else

Select a node that has the smallest objective function value

Endif

Step 3. Solve the corresponding linear relaxation and obtain the optimal solution

Step 4. *If* current node is infeasible or worse than upper bound, *then*
Remove the current node from the tree and *goto* Step 2.
Endif

Step 5. *If* Optimal solution is integer, *then*
Update the upper bound and remove it from tree and *goto* Step 2.
Endif

Step 6. *If* Optimal solution is not integer, *then*
Construct a feasible solution from existing solution and update the upper bound.
Endif

4.3. VRPTW using Exact method

4.3.1. Lagrangian Relaxation using subgradient approach

Here, an approach based on Lagrangian relaxation is presented to solve the bi-objective proposed formulation described in Section 4.1. The main approach includes the relaxation of the integrity constraints which gives the relaxed problem which is easier to solve and it gives the lower bound on the original problem. Then, a feasible solution to the problem is constructed which gives the upper bound. Here, Sub gradient method is used for the non-differentiable constraints which help to adjust the Lagrange multipliers so as to reduce the constraint violation, and this procedure will update the Lagrange multipliers. This process is repeated until some stopping condition is not met.

4.3.2. Algorithm

1. Initialize parameters Lower Bound (LB) = -∞, Upper Bound (UB) = +∞, lagrangian multiplier $\lambda_i = 0, \forall i$ constraints. Primal is used to obtain better UB.
2. Repeat
3. Lower Bound is evaluated

Z_{LB}=getObjectiveValue()

If (LB<Z_{LB})

Modify LB=Z_{LB}

Z_{LB} is the lower bound on the optimal objective value. It is obtained by computing the model relaxing the required constraint which is marked as "hard".

4. If lower bound is improved in previous step apply primal heuristics else goto step5
5. Set UB=primal heuristic solution, Compute the UB

   Z_{UB}= getObjectiveValue()

   If (UB>Z_{UB})

   UB=Z_{UB}
6. If UB< (LB + tolerance), terminate algorithm, found optimum else go to Step 6.
7. Apply gradient search λ_i
8. If norm=0, terminate found optimum(LB is optimum in this case)
9. Until the iteration count is reached
10. Output the results.

Figure 4.3 suggests the proposed approach. It depicts the working of the solution approach. The following are the various conditions required for the lagrangian relaxation:

A. Computation of Lower Bound

The technique used to find linear bounds is linear programming relaxation. Here, the mixed integer programming formulation of the problem is considered and the integrity requirements on the variables are relaxed. Here, Constraint (4.2) is relaxed which satisfies that each customer must be serviced with the Lagrangian multiplier, thus obtaining the corresponding Lagrangian relaxation problem.

$$Minimise \sum_{k \in C'} \sum_{(i,j) \in E} c_{ij} X_{ijk} + \sum_{i \in C} \lambda_i \left(1 - \sum_{k \in V} \sum_{j \in C'} X_{ijk}\right) \quad (4.12)$$

We rewrite the above equation as:

$$Minimise \sum_{k \in C'} \sum_{(i,j) \in E} (c_{ij} - \lambda_j) X_{ijk} + \sum_{i \in V} \lambda_i \quad (4.13)$$

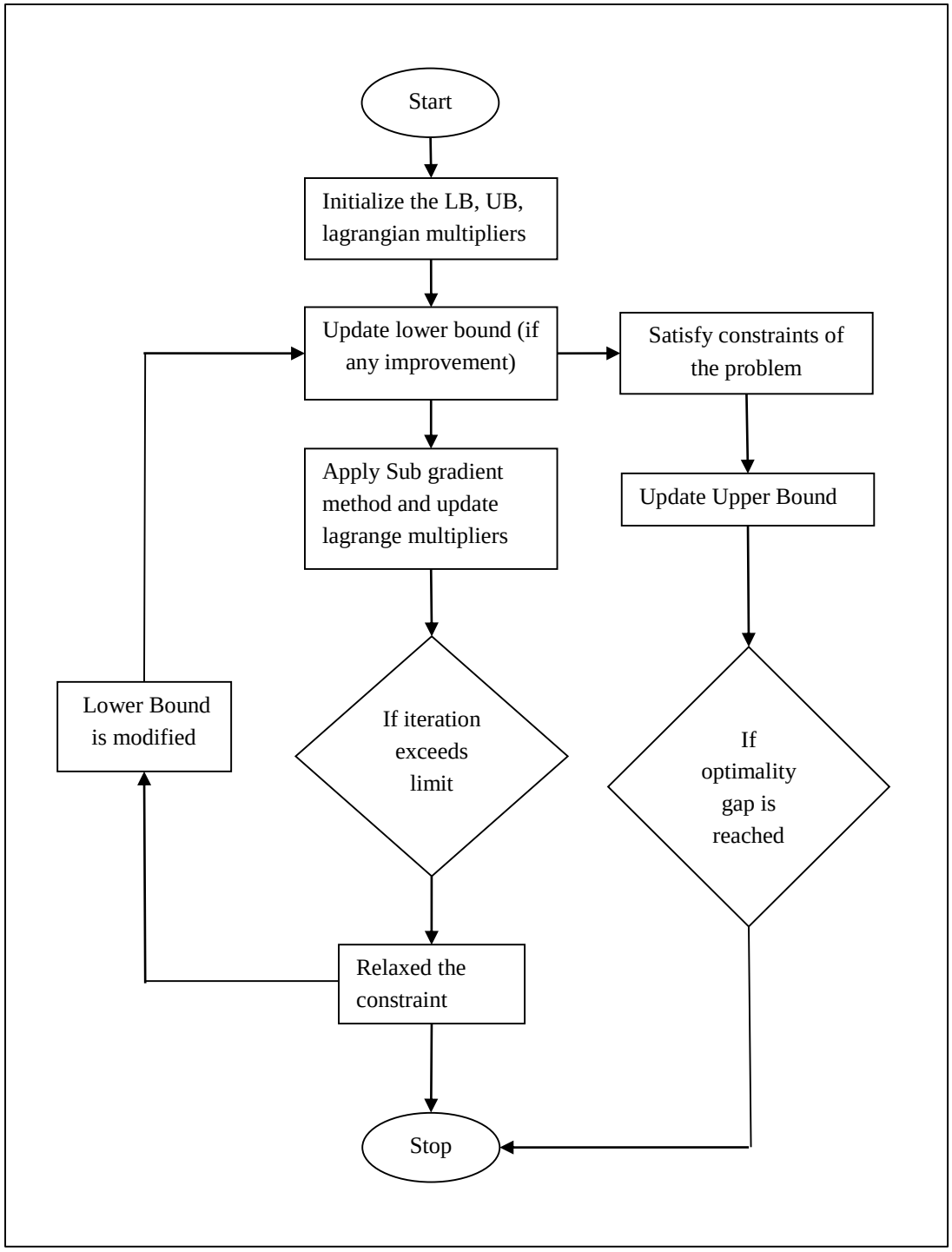


Figure 4.3. Flowchart of the Lagrangian relaxation

B. Computation of Upper Bound

At each iteration, feasible solution to the problem is calculated by finding feasible solution by satisfying the constraint of the problem. It depends on the results of the relaxed problem generated by relaxing the required constraint.

C. Sub gradient method

It is an iterative method which involves updation of multipliers in some systematic manner. This procedure helps to maximize the lower bound obtained from the relaxed problem.

The basic steps are

1. Let π be the user defined parameter with value in range (0, 2]. Here, in our problem we set this parameter value equal to 1.
2. Solve the relaxed problem for the lower bound to get solution Z_{LB}.
3. Evaluate subgradients for the constraint (2)

$$G_i = 1 - \sum_{k \in V} \sum_{j \in C'} x_{ijk}, \ \forall i \in C \tag{4.14}$$

4. Calculate step size

$$step = \pi (Z_{UB} - Z_{LB}) / \sum_{i=1}^{m} (G_i)^2 \tag{4.15}$$

This step size depends upon the gap between the upper bound and lower bound.

5. Update the multipliers λ_i

$$\lambda_i = \max(0, \lambda_i + step \times G_i) \tag{4.16}$$

D. Stopping criterion

The algorithm will terminate in any of the condition met:

- Optimality Gap: This happens when gap between the lower and upper bound gets close to zero
- Iteration limit: When the limit of the iteration is reached to the specified value. Here the iteration limit is generally set to be 100.

- Step size: When the constant step size gets close to zero.

4.4. VRPTW using Metaheuristic technique

Due to the complexity of VRP family problems, exact methods become useless and intensive where large problem instances are considered. Therefore, the approximate algorithm are useful in those situations. Metaheuristic techniques are used due to easy implementation. Here firefly algorithm, which has attracted a lot of attention in recent past implemented here.

4.4.1. Firefly Algorithm

4.3.1.1. Classical Firefly Algorithm

Fireflies are small winged insects capable of producing a light flashes in order to attract mates. Firefly Algorithm (FA) is Nature-inspired metaheuristic algorithm idealizing the flashing characteristics of fireflies. It is based on some assumptions

- Each firefly emits flashes to attract other fireflies therefore firefly with less brightness will move towards the brighter one.
- Attractiveness is proportional to the brightness and, attractiveness and brightness are inversely proportional to distance.
- No fireflies is capable of attracting the brightest firefly, thus it moves randomly.

Fireflies' attraction is based on two major factors: its brightness and attraction. Brightness is judged by the objective function value of the fireflies' position. More the brightness of the firefly, the better is the location. Attraction is proportional to the brightness. The brighter firefly has more attraction thus it can attract the fireflies with lesser. In case of brightest firefly, a firefly will just move randomly. It has no firefly with higher brightness than itself.

Psuedo Code of the Classical Algorithm

INPUT : n *fireflypositionvectors*, $\alpha, \beta, \gamma, t = 1$, *maxgeneration*;

OUTPUT : *A vector of fireflywith highestvalueof intensity*

ALGORITHM :

Define Objective function f(x), $x=(x_1,...,x_d)^T$

Initialize the firefly position vectors x_i $(i=1,2,...,n)$

Define light absorbtion coeffiecient γ

for each firefly x_i *in the population*

Initialise light intensity I_i

while$(t < maxGeneration)$

for $i = 1:n$ (*where n is number of fireflies*)

for $j = 1:n$

if $(I_i < I_j)$, *Move fireflyi towards j*

endif

attractiveness is varied with distancer via $\exp(-\gamma r)$

Evaluate new population vectors and compute updated light intensity

end for

end for

increment t

end while

find globalbest

Print Results

A. Key parameters

1) Distance between fireflies: Distance between two fireflies can be calculated as

$$r_{ij} = \left\| x_i - x_j \right\| \tag{4.20}$$

The distance r_{ij} is the calculated distance between two fireflies x_i and x_j. This distance can be calculated with any of the mentioned distance measures.

2) Light Intensity and Brightness: In simplest term, the light intensity varies with inverse square law as

$$I = \frac{I_s}{r^2} \tag{4.21}$$

where I_s is the intensity at the source. But when a medium with light absorption coefficient γ, I varies with distance r.

$$I = I_o e^{-\gamma r} \tag{4.22}$$

where I_0 is associated with objective function value at $r = 0$ .This also signifies that fireflies brightness decreases with the distance increase and the absorption coefficient increase.

3) Attractiveness: Attractiveness of a firefly is proportional to the light intensity and can be expressed as

$$\beta = \beta_o e^{-\gamma r^2} \tag{4.23}$$

where β_o is the attractiveness at $r = 0$. r is the distance between two fireflies and γ is the light absorption coefficient.

4) Position update: In the case of the intensity difference of two fireflies with firefly i having more intensity than firefly j, the firefly i is attracted to the firefly j and the movement is determined by

$$x_i = x_i + \beta_o e^{-\gamma r^2}(x_j - x_i) + \alpha \varepsilon_i \tag{4.24}$$

where the second term is due to the attractiveness of firefly i towards j. The third term contains α as the randomization parameter and ε_i is the vector of random numbers which can be substituted by $rand - 1/2$ where rand is the random number generator generating numbers in the range $[0,1]$.

The movement of the fireflies are the biased random walk towards the brighter firefly.

B. Application to the Vehicle Routing Problem with time windows

Below are the steps to be followed to apply the Classical Firefly Algorithm to the Vehicle Routing Problem with time windows.

Step 1. Initialization of Algorithm parameters

The number of fireflies n, the largest degree of attraction β_o, the degree of light attenuation γ, step factor α , and the maximum number of iterations are initialized**.** Number of fireflies can be varied in the range of 15 to 40, β_o has values in range (0, 1). In theory $\gamma \in [0,\infty)$, however γ =1 is used in of the implementation, α takes on values in range (0,1).

Step 2. Generation of population space randomly

Take the dimension of firefly as $n+V-1$. Initialize the position vectors of the firefly randomly in the range of 1 to $n+V-1$.

Step 3. Calculation of distance between the firefly's position vectors. Any one of the method can be used here for distance calculation.

Step 4. Updation in randomization parameter

The randomization parameter α is updated in each iteration as

$$\alpha_{new} = a*(1-(1-\frac{10^{-4}}{9})^{1/\max Gen}) \tag{4.25}$$

Step 5. Evaluation of the best firefly

Evaluate the relative brightness I . Based on the brightness of the fireflies, the firefly with highest brightness is regarded as best firefly.

Step 6. Updation in position vectors

Update the postion vectors of the less bright firefles by using equation (4.24).

Step 7. Updation in position of brightest firefly

Here the position of brightest firefly is updated by a factor of $rand$-1/2.

Step 8. Repeat Steps 3-7 untill the maxGen value is reached. Figure 4.4 gives the flowchart for the Classical Firefly algorithm.

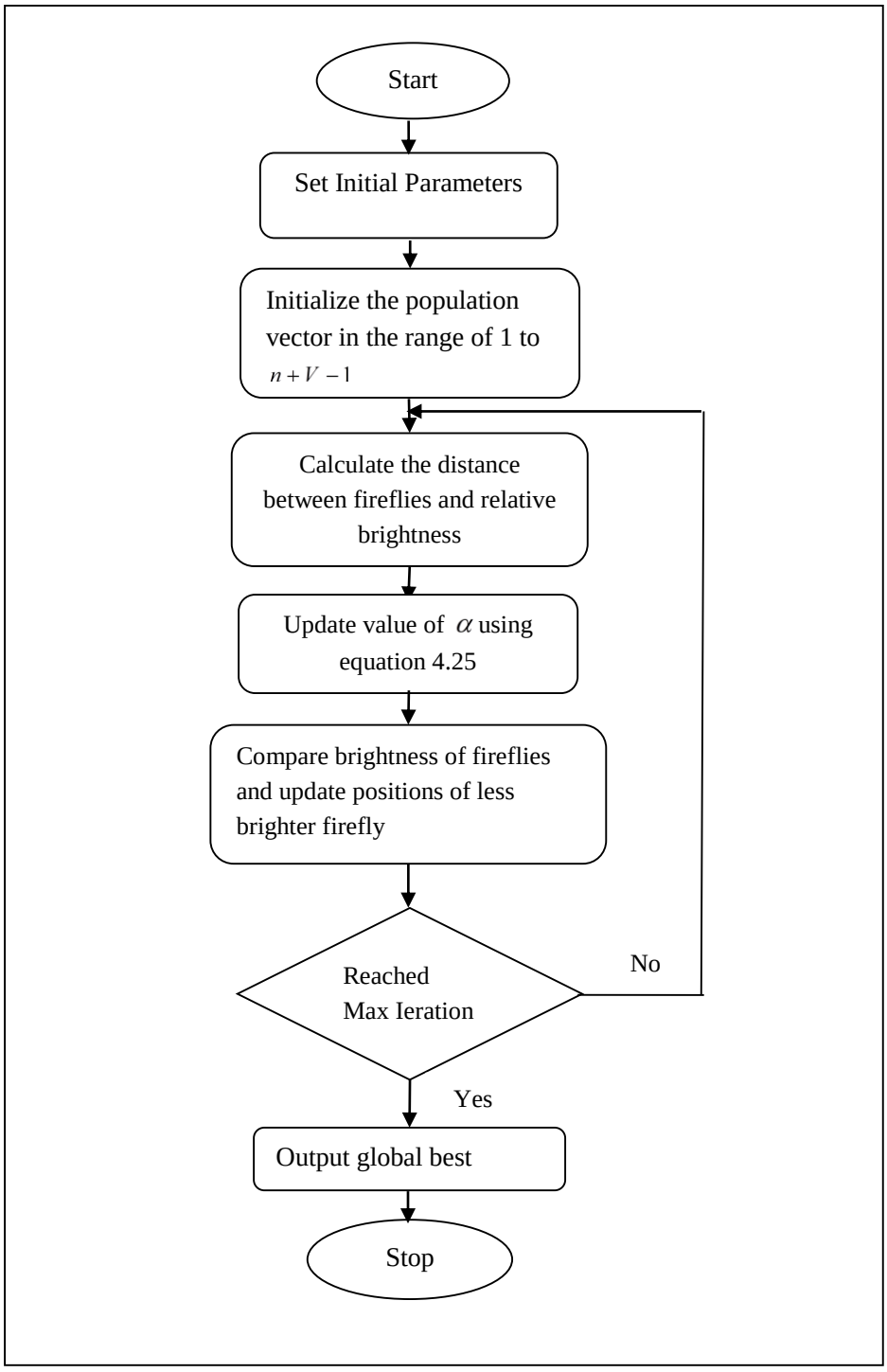


Figure 4.4. Flowchart of the Classical Firefly Algorithm

4.4.1.2. Distance measures

Here four different distance measures are discussed to calculate the distance between the position vectors of fireflies. The distance measures are discussed as below:

1) Euclidean Method

   The Euclidean method also known as Cartesian method is the most used distance method. The Cartesian distance, $D_{Cartesian}$ between two points x_i and x_j can be computed as [57]:

$$D_{Cartesian} = \sqrt{\sum_{d=1}^{D} \left| x_{id} - x_{jd} \right|^2} \tag{4.17}$$

   Here x_{id} and x_{jd} is the d_{th} dimension of the points.

2) Hamming Method

   Hamming Distance between two points is defined as number of unequal corresponding vectors in the position values [57].

   e.g. Let x_i and x_j are two points with position vectors as

   $x_i = (3,5,6,7,2,1)$

   $x_j = (4,5,7,2,2,1)$

   Hamming Distance between these two points is 2.

3) Chebyshev method

   The Chebyshev distance method is defined as the maximum among the absolute differences between the corresponding vectors in the position values. The Chebyshev distance, $D_{Chebyshev}$ between two points x_i and x_j with dimension D can be computed as [57]:

$$D_{Chebyshev} = \max_{1 \le d \le D} \left(\left| x_{id} - x_{jd} \right| \right) \tag{4.18}$$

4) Brute-Curtis method

   The Brute-Curtis distance method is computed as the maximum among the absolute differences divided by the summation. The Brute-Curtis distance, $D_{Brute-Curtis}$ between two points x_i and x_j can be computed as [57]:

$$D_{Brute-Curtis} = \frac{\sum_{d=1}^{D} |x_{id} - x_{jd}|}{\sum_{d=1}^{D} (x_{id} + x_{jd})} \quad (4.19)$$

4.3.2. Modified Firefly Algorithm

4.3.2.1. Motivation

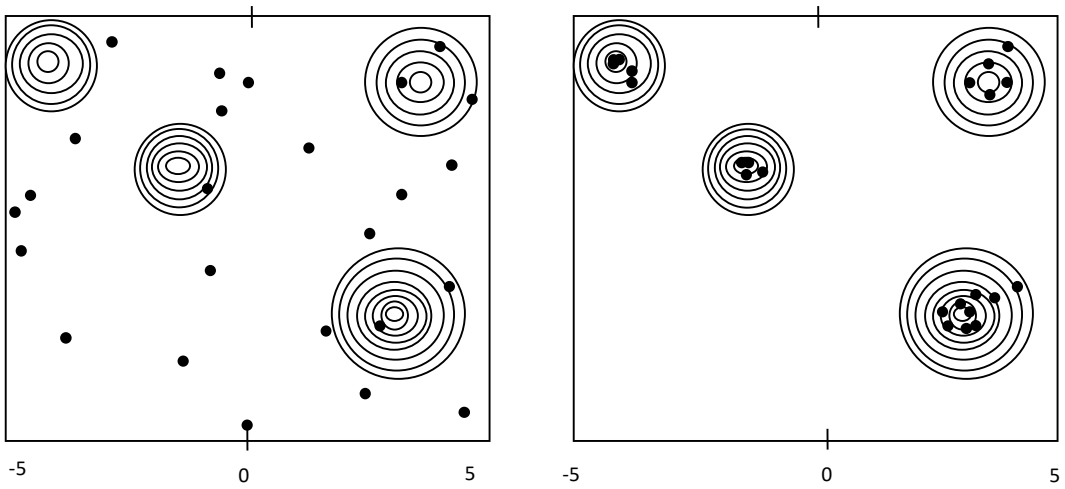
The mentioned researches in literature have generally focused only on the minimization of total transportation cost. Here, the main concern is the bi-objective optimization considering the minimization of the total transportation cost and number of vehicles. Firefly Algorithm has been very efficient Nature inspired algorithm. Hence, a modification to the firefly algorithm has been proposed to work on the bi-objective VRPTW.

4.3.2.2. Proposed Approach

Here, a modification to the classical firefly is suggested. Chebyshev method has been used for distance calculation rather than Cartesian and hamming distance method which were consistently used in the literature. The Chebyshev method was analysed to give better results than the other method of distance calculation in the previous section. Another modification suggested is the reduction of problem space at each iteration. The problem space is reduced to the limit the space around the brightest firefly. Another modification is the updation of position of brightest firefly only if there is improvement in the brightness.

The Firefly algorithm starts by initializing a vectors of fireflies position which follows some iteration steps iteratively so to move the firefly towards the brightest firefly. Brightness of the firefly is associated with the objective function of the problem. Therefore, it works on finding the brightest firefly in the region which gives the best value for the objective function. To apply Firefly algorithm for the VRPTW, the relationship between the firefly positions and the problem solution need to be established. A method which converts the firefly positions to the problem solution is here, called the decoding method. The movement of fireflies during the firefly algorithm is described in Figure 4.5. The left

section of the image shows the initial positions of the fireflies while the right section of the image shows the positions after the 30 iterations.



(a) (b)

Figure 4.5. Progress towards iterations (a) the initial locations of 25 fireflies and (b) the final locations of 25 fireflies after 30 iterations

Psuedo Code of the Modified Firefly Algorithm

INPUT : *n firefly positionvectors*,$\alpha,\beta,\gamma,$,*maxgeneration*;

OUTPUT : *A vector of fireflywith highestvalueof intensity*

ALGORITHM :

Initialise the set parameters α, β, γ .

Initialisethe N fireflies positionvectors

Repeat

for $i = 1:N$ (*where N is numberof fireflies*)

Decodethe positionvector intoroutes

Evaluatethe fitness functionsand storethebest firefly

Update the randomisation parameter

for $j = 1:N$

for $j = 1:N$

---

if $(I_i < I_j)$

attractiveness is varied with distance r via $\exp(-\gamma r)$

Update the position of firefly i towards j

Update the position of brightest firefly

end if

end for

end for

Modify infeasible solution

Reduce population space

Until the max iteration reached

Print results

4.3.2.3. Application to the Vehicle Routing Problem with time windows

Step 1. Initialization of the firefly's position vectors

In the initial step, N firefly vectors consisting of $n+V-1$ dimension is initialized which can match with final distribution solution. Uniform random values are assigned to each of the dimension in range $(1, n+V-1)$ and no value is repeated in single vector. The valued sequence of each vector conveys the information of distribution order on the total routes derived.

Step 2. Decoding the initialized population to problem solution

Converting the firefly positions to the problem solution is termed as Decoding. Decoding can be done by following this example.

Example: Suppose there are seven customers and four vehicles

Let $x = \{6,1,3,10,2,8,5,7,4,9\}$ be a position vector and it results to solution of total routes as $0 \to 2 \to 5 \to 3 \to 9 \to 7 \to 1 \to 8 \to 6 \to 10 \to 4 \to 0$, where 0 is considered as depot locations and 8,9,10 are called virtual centres signifying the different parts of complete route.

Here the set of routes are

$0 \to 2 \to 5 \to 3 \to 0$

$0 \rightarrow 7 \rightarrow 1 \rightarrow 0$

$0 \rightarrow 6 \rightarrow 0$

$0 \rightarrow 4 \rightarrow 0$

These four routes are traversed by corresponding four vehicles. Here, the two zeroes are added to mark the start and end of the route. Therefore, two virtual centres are not allowed to be adjacent and the corresponding vector is not considered for fitness evaluation.

Step 3. Evaluation of the fitness function

Decoded set of routes are evaluated for the minimum distance and the minimum number of vehicles by the calculating following functions

$$F1 = \sum_{k=1}^{V} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} d_{ij} \tag{4.27}$$

F1 is the distance calculated for each firefly. Here n is the size of each route and total distance is summed for each vehicle.

$$\mathrm{F2} = \text{number of number decoded}$$

Here F2 gives the minimum number of vehicles used. Generally every vehicle traverse one routes so decoded number of routes gives the minimum number of vehicles used.

Step 4. Evaluation of best firefly

The firefly with minimum value of fitness function is considered as the best firefly. Best firefly is the firefly with maximum brightness and it attracts the other fireflies in the surroundings. Decoding this firefly can give the final output.

Step 5. Reduction of randomization parameter

At each iteration of the method, value of α is updated to reduce the randomness of new population. α is the randomization factor, hence reducing the parameter at each iteration reduces the region of expansion of new population

$$\alpha_{new} = \alpha_{old} * (1 - (1 - \frac{10^{-4}}{9})^{1/\max Gen}) \tag{4.29}$$

Step 6. Calculation of distance between the firefly position vectors.

Here, the Chebyshev method analysed to be best among the four different distance measures is used for distance calculation between two firefly's position vectors. $D_{Chebyshev}$ between two points x_i and x_j having D dimension each can be computed as:

$$D_{Chebyshev} = \max_{1 \le d \le D}\left(\left|x_{id} - x_{jd}\right|\right) \tag{4.30}$$

Here, distance is the defined as the maximum value among the absolute differences between the corresponding vectors in the position values.

Step 7. Calculation of firefly attractiveness

Attractiveness of a firefly is proportional to the light intensity and can be expressed as

$$\beta = \beta_o e^{-\gamma r^2} \tag{4.31}$$

where β_o is the attractiveness at $r = 0$. r is the distance between two fireflies and γ is the light absorption coefficient.

Step 8. Updation of firefly's position vectors

In case of the intensity difference between the firefly i and j, $I_i < I_j$, the firefly i is attracted to the firefly j and the movement is determined by

$$x_i = x_i + \beta_o e^{-\gamma r^2}(x_j - x_i) + \alpha\varepsilon_i \tag{4.32}$$

where the second term is due to the attractiveness of firefly i towards j. The third term contains α as the randomization parameter and ε_i is the vector of random numbers which can be substituted by $rand - 1/2$, where rand is the random number generator generating numbers in the range [0,1] . The movement of the fireflies are the biased random walk towards the brighter firefly [60].

In modification to the classical firefly, here we also update the best firefly by random value $rand - 1/2$ and check if there is any improvement in the position by evaluating the fitness function. If an improvement occur, then replace the best firefly otherwise, discard and keep the original one.

Step 9. Modification of the infeasible solution

The position vector of the firefly keeps on changing in the iterative process. Each firefly keeps on following the brightest firefly in the group. The updated position vectors does not always guarantee the uniformity of values in the range $(1, n+V-1)$ even after the rounded treatment. Therefore, to maintain the effectiveness of the solutions, we identify the duplicated positions in the vector and change them to cover the range. Hence only one of the duplicate value is kept and other is replaced with the non-covered numbers in the required range.

Step 10. Reduction of problem space

Here, modification of reducing population space is added. At each iteration, number of firefly is reduced to the space around the best firefly. The distance to the best firefly is proportional to the potential to be nominated as best firefly. Hence, the problem space can be easily reduced without the loss of optimality. Moreover, reducing the space will reduce the computation time.

Step 11. Checking the Stopping criterion

- If the Maximum Number of iteration is reached, then termination occurs. Here, the number of MaxIteration is fixed to 50.
- While reducing the number of fireflies in each iteration. If number of fireflies become 1, then termination should occur.

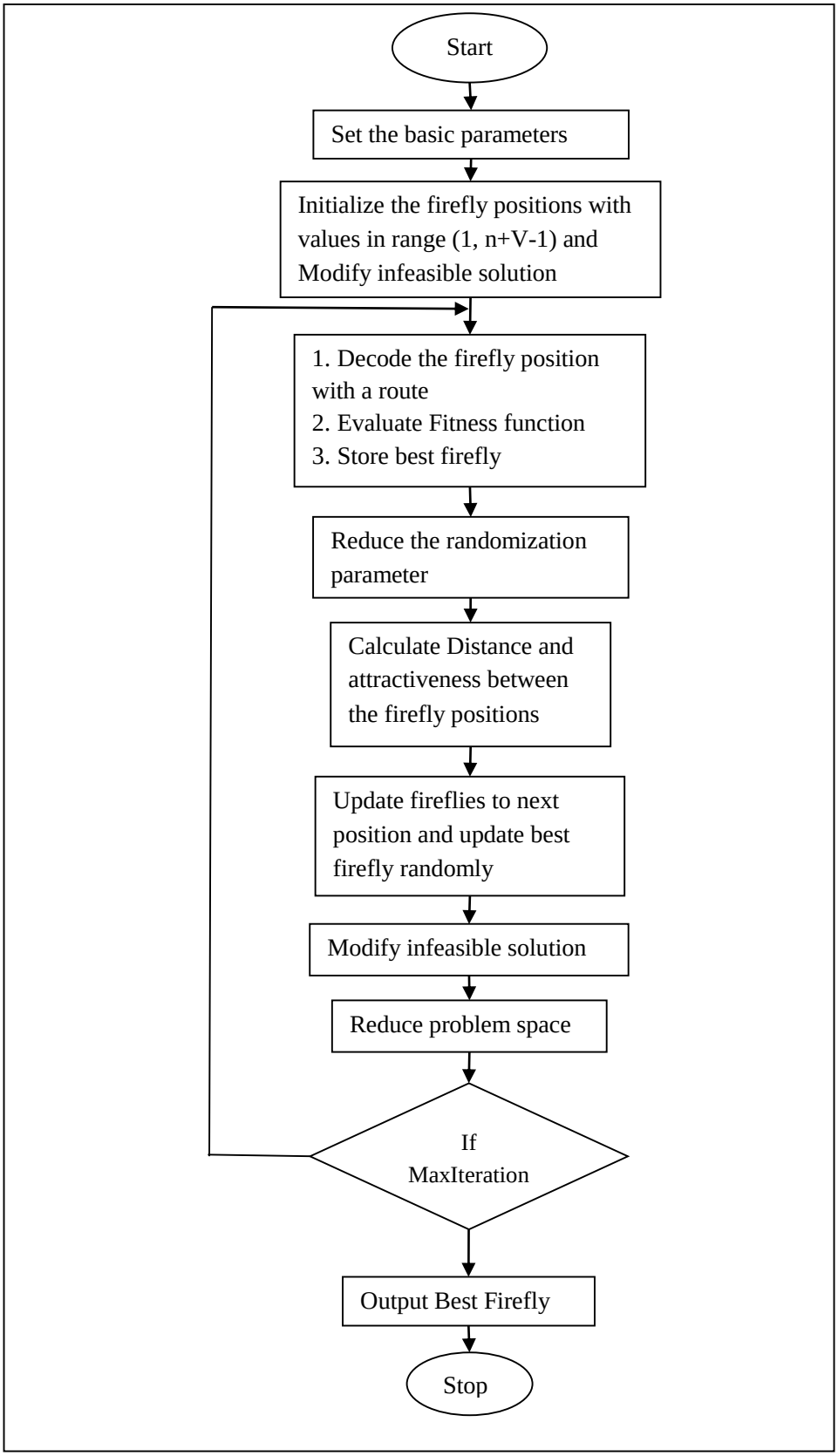


Figure 4.6. Flowchart of the Modified Firefly Algorithm

Chapter 5
Experimental Results and Analysis

5.1 Tool Used

The term optimisation consists of maximizing and minimizing a function by selectively choosing input values from within defined set of values and then computing the value of the function. Optimization software provides better design and development of optimization solutions for real-life problems. The software generates different solutions under different constraints. Here IBM CPLEX is used for the purpose.

IBM CPLEX is one of the tools widely used to solve combinatorial optimization problems. It has a concert technology that provides interfaces to C++, C# and Java languages. It is accessible through independent modeling systems such as AMPL, and TOMLAB. It is also recognized as a constraint solving toolkit suitable for solving optimization models. It uses inbuilt procedures to solve the mixed integer programming in short time. It can be used to solve a variety of different optimization problems in a variety of computing environments. IBM CPLEX is an exact solver that uses mixed integer programming to search the desired solutions.

5.2. MIP approach: Results and Discussion

In this section, the experimentation is designed to evaluate the performance of proposed approach on two well-known benchmark instances of Solomon [58]. The algorithm was coded in IBM CPLEX and implemented on a Core i7 Intel Processor under Windows 7 with 4 GB RAM.

5.1.1 Test Instances

The proposed approach is tested on two test instances of vehicle routing problem with time windows. The used test instances are taken from Solomon's VRPTW instances described in Appendix A. The test instances are R101 and C101. The first test instance namely R101

that belongs to class with randomly generated customers. This class is solvable upto 33 customer nodes. The second test instance namely C101 that belongs to class having clustered customer locations. This class is solvable upto 60 customer nodes. The objective of proposed approach is to minimize the number of vehicles and carrying capacity of each vehicle should be 200 units.

5.1.2 Performance Evaluation

Table 5.1 shows the performance evaluation of proposed approach on test instances R101. The six different cases are generated from R101 instance. The number of customers is varying from 5 to 33. The computed distance, used number of vehicles and computational time are taken for performance comparison. As seen from Table 5.1, the computational time increases with increase in number of customer node. The optimum distance and vehicle used is also increase with increase in customer nodes.

The performance evaluation on test instances C101 is presented in Table 5.2. The seven different cases are generated for C101. The number of customer nodes is varying from 5 to 60. It has been observed from Table 5.2 that the number of vehicle used is one for number of customer nodes 5 and 10. For number of customer nodes 20 and 25, the optimum vehicles used are 3. The number of vehicle used, objective function value, and computation time increase with increase in number of customer nodes.

Table 5.1. Performance evaluation of proposed approach on test instance R101.

Instance	**Number of customer**	**Objective Distance**	**Minimized Number of Vehicles**	**Average time (in seconds)**
R101	5	156.35	2	0.76
R101	10	269.54	4	1.92
R101	15	383.83	3	1.99
R101	20	511.26	6	2.45
R101	25	618.34	6	4.59
R101	33	713.58	11	11.59

Table 5.2. Performance evaluation of proposed approach on test instance C101.

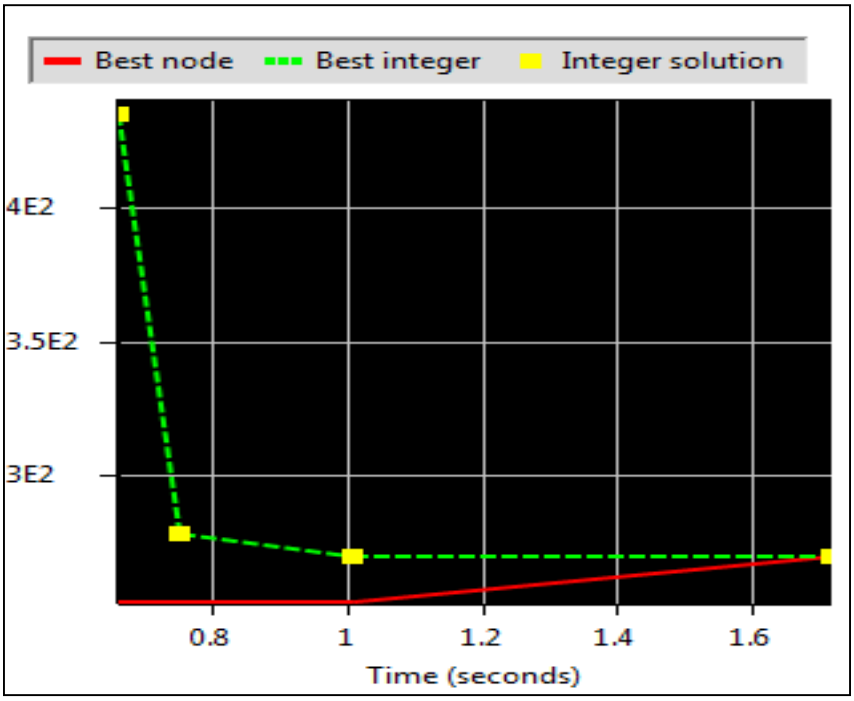
Instance	Number of customer	Objective Distance	Minimized Number of Vehicles	Average time (in seconds)
C101	5	42.42	1	0.69
C101	10	58.34	1	1.20
C101	15	142.17	2	2.16
C101	20	175.39	3	2.89
C101	25	191.83	3	5.22
C101	50	363.28	5	19.49
C101	60	470.13	6	36.92

5.1.3 Convergence Analysis

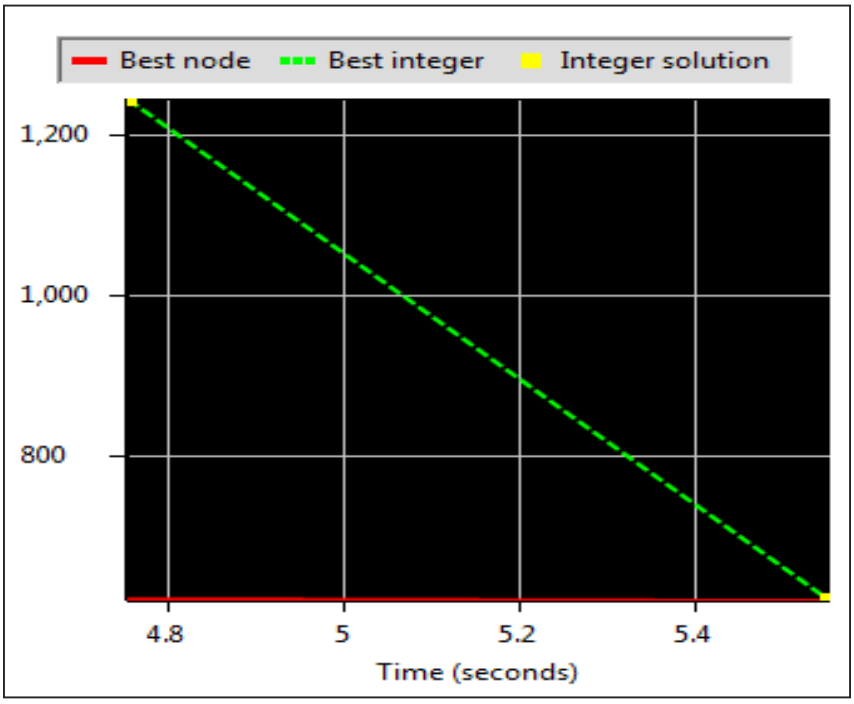
Figures 5.1 and 5.2 show the convergence curves obtained from proposed model using R101 and C101 test instances respectively. In these figures, the yellow point indicates a node where an integer value has been found. The green line represents the evolution of best integer value computed. The red line gives a bound on the final solution. These figures are drawn between objective function value and time.

It has been observed from figures that the proposed model explore the promising regions for solutions. Thereafter, it converges towards the best solution. Figures 5.1 (a) and 5.1 (b) show the convergence curve for 10 and 25 customer nodes. As seen from Figure 5.1 (a), the slope of convergence curve is varying throughout the process whereas the slope of convergence curve is same throughout the process for 25 customer nodes. From Figures 5.2 (a) and 5.2 (b), it has also been observed that the slope of convergence curve is same throughout the process for both 10 and 25 customer nodes.

Therefore, mentioned results depict that the proposed model provides better convergence towards the optimal solution. The performance of proposed model is least affected for above-mentioned customer nodes.

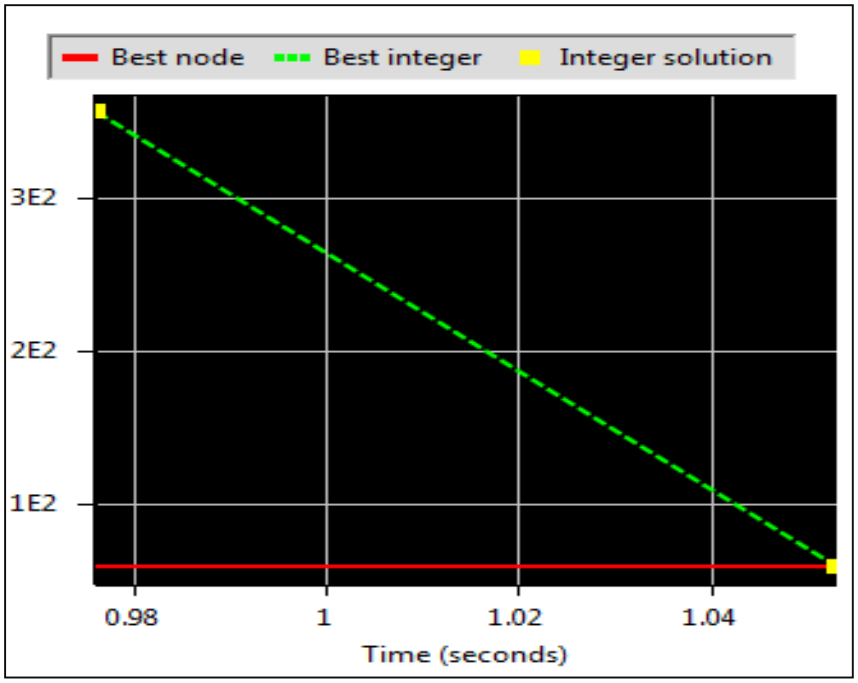


(a)

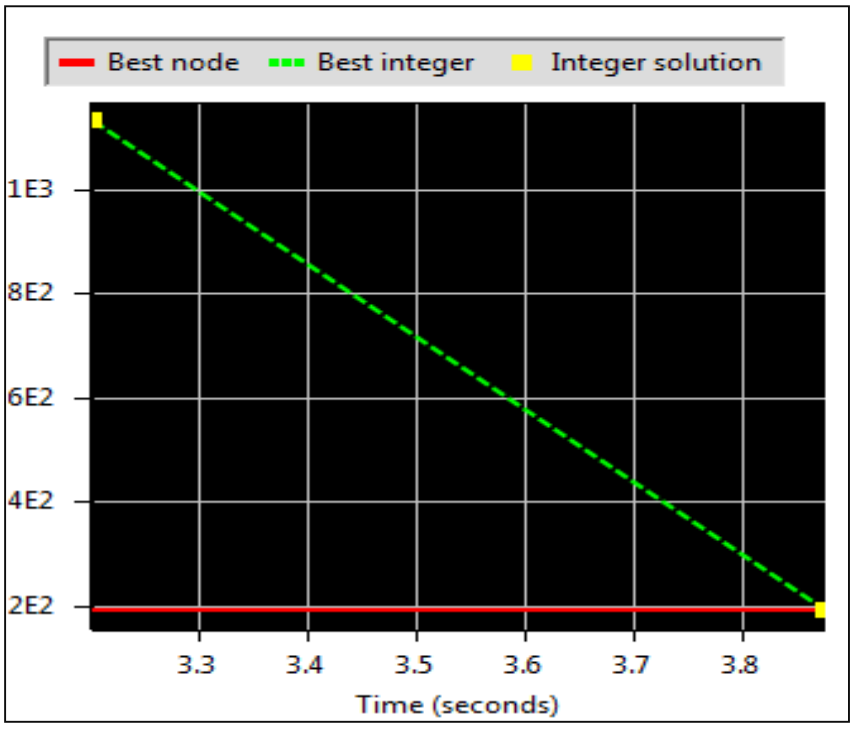


(b)

Figure 5.1. Convergence curve obtained from CPLEX (a) 10 customer nodes (b) 25 customer nodes for test instance R101.



(a)



(b)

Figure 5.2. Convergence curve obtained from CPLEX (a) 10 customer nodes (b) 25 customer nodes for test instance C101.

5.1.4 Sensitivity Analysis

The effect of number of customer has been investigated on computational time and vehicle used is studied. Figures 5.3 and 5.4 show the performance of proposed approach over R101 and C101 instance respectively. For R101 test instance, the computational time greatly increase with increase in number of customers. The number of vehicle used also increases with increase in the value of customer nodes. The result reveal that the number of vehicles used is six for both 20 and 25 customer nodes.

For C101 test instances, the performance of proposed approach is greatly reduced with increase in number of customers. The results obtained from Figure 5.3 depict that the number of vehicles used and average computation time increase with increase in customer nodes. The optimum number of vehicle used is one for 5 and 10 customer nodes. It can also be observed that the minimum number of vehicle used is three for 20 and 25 customer nodes.

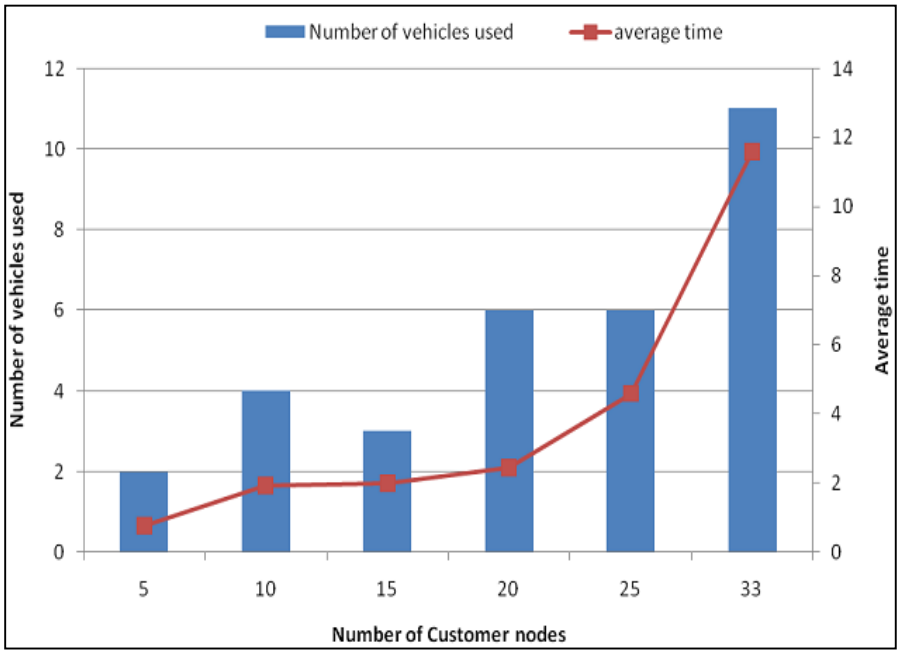


Figure 5.3. Comparison between customer nodes and vehicles used with average time on secondary axis for R101 instance.

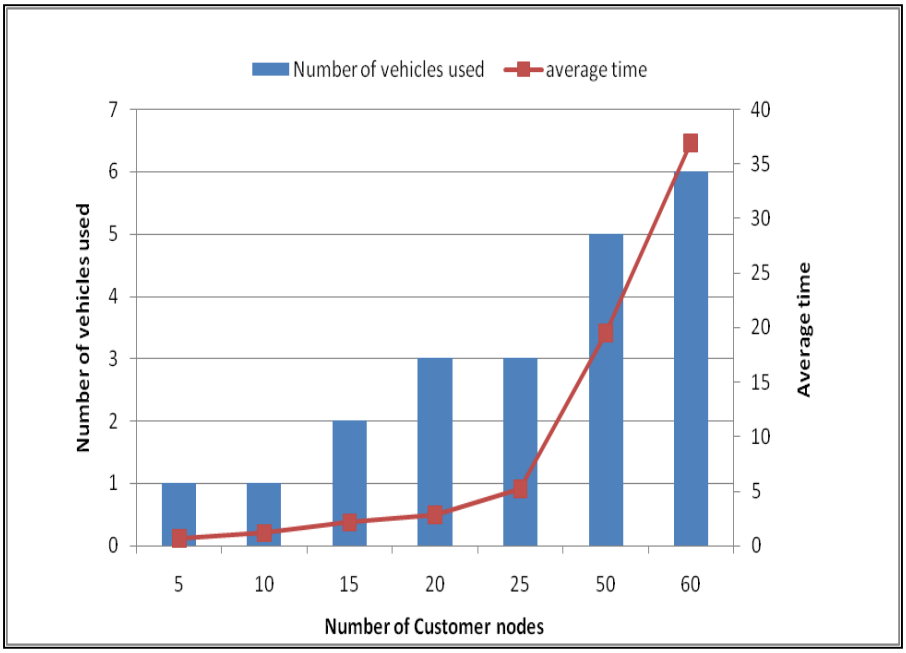


Figure 5.4. Comparison between customer nodes and vehicles used with average time on secondary axis for C101 instance

5.3. Exact method for VRP: Results and Discussion

The model specified in Section 3 is developed and implemented in CPLEX 12.6 on a PC clocked at 2.2 GHz under the operating system Windows 7.

5.3.1. Dataset used

Here, the model is tested on series of benchmark problems R101-R112 from the Solomon instances [58] which belong to the class of randomly generated customers Solomon's benchmark test instances are described in Appendix A. Here, Table 5.3 shows the computational results obtained from R1 class instances. The objective of the problem is minimizing the transportation cost and to minimize number of vehicle should be used from 25 vehicles, each vehicle having 200 capacity.

5.3.2. Performance Evaluation

It is observed from Table 5.3 that first column gives the instance considered, second column records the Lower bound objective value, third columns gives the optimal solution

value, fourth column records the number of vehicle used of the total vehicles, fifth column calculates the gap to the optimal solution and iterations of the problem. Sixth column and seventh column records iterations and computation time of the problem. Fig. 3 shows the statistic graph of the algorithm on R101 for 25 customer nodes.

Table 5.3. Results of the algorithm

Problem Instance	**Lower Bound (LB)**	**Optimal Solution**	**Vehicle**	**Gap (%)**	**Iterations**	**Time**
R101.25	618.34	617.1	6	-0.20	42	1.24
R101.50	1046.2	1044.0	11	-0.210	70	1.98
R101.100	-	1637.7	-	-	220	4.34
R102.25	547.8	547.1	7	-0.12	63	1.25
R102.50	909	909	11	0	137	1.34
R102.100	1468.2	1466.6	17	-0.10	240	20.87
R103.25	453.2	454.6	4	0.30	58	0.87
R103.50	770.8	772.9	8	0.27	270	7.50
R103.100	1207.9	1208.7	13	0.06	660	8.39
R104.25	416.9	416.9	4	0	71	1.11
R104.50	625.4	625.4	5	0	689	10.83
R104.100	971.5	971.5	11	0	1243	10.87
R105.25	530.5	530.5	5	0	45	3.21
R105.50	899.3	899.3	9	0	143	4.54
R105.100	1355.3	1355.3	15	0	520	9.59
R106.25	460.2	465.4	3	1.11	77	1.11
R106.50	790.87	793	5	0.26	128	1.53
R106.100	1226.33	1234.6	12	0.66	2983	12.35
R107.25	425.2	424.3	4	-0.21	70	1.25
R107.50	713.8	711.1	7	-0.37	376	1.34
R107.100	1068.3	1064.6	11	-0.34	6580	20.8
R108.25	396.66	397.3	4	0.16	115	1.11
R108.50	615.34	617.7	6	0.38	15643	1.53
R108.100	-	-	-	-		-
R109.25	441.1	441.3	5	0.045	23	0.98
R109.50	777.33	786.8	8	1.20	567	5.21

R109.100	1140.32	1146.9	13	0.57	10945	8.45
R110.25	444.1	444.1	4	0	119	1.3
R110.50	694.80	697.0	7	0.315	110	3.7
R110.100	-	1068	-	-	-	-
R111.25	425.43	428.8	5	0.78	77	1.11
R111.50	706.2	707.2	7	0.14	998	1.53
R111.100	-	1048.7	12	-	-	-
R112.25	390.5	393	4	0.63	123	2.8
R112.50	620.8	630.2	6	1.49	15673	20.87
R112.100	-	-	-	-	-	

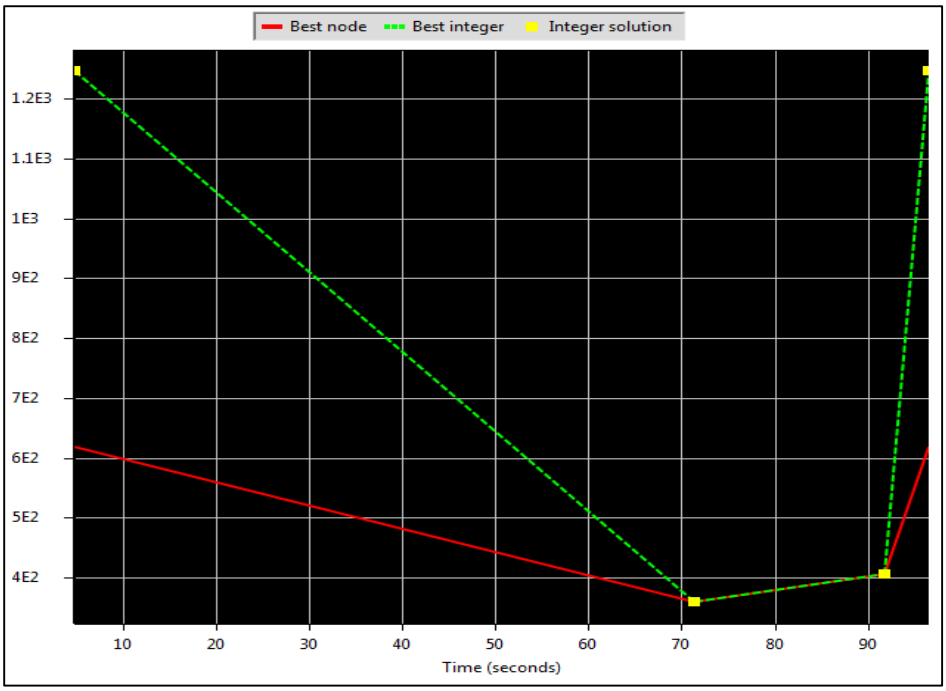


Figure 5.5. Statistics graph for thr R101 instance with 25 customers

The results generated are quite encouraging and better than optimal solutions in most of the cases. In some cases, the results are not computable and shown by (-). Here number of vehicle is also reduced. The results are obtained in short amount of time. Here optimality gap is calculated so to indicate the quality of solutions. Positive value of optimality gap

shows the better values of solution than the optimal values. Optimality gap is used to find tighter bounds.

5.4. Firefly Algorithm for VRP: Results and Discussion

5.4.1 Dataset used

Instances of the well-known Solomon's benchmark test problems described in Appendix A [58] are used here with different number of customer nodes. R101 and C101 are the two test instances where R101 belongs to the class with randomly generated customers and C101 has clustered customer locations. Here R101 and C101 are solved for 25 and 50 customer nodes.

5.4.2. Parameter setting

The parameters are set as follows

Table 5.4. Parameter setting

Number of fireflies	15
Step factor α	0.5
Maximum attractiveness β_0	1
Light absorption coefficient γ	1

These parameters are evaluated by trial and error method.

5.4.3. Performance Analysis

The algorithm is implemented in CPLEX 12.6.3 on a PC clocked at 2.2 GHz under the operating system Windows 7. Here the distance between the firefly positions is calculated with the help of the distance metric discussed above and the results are analyzed. The mean % error is defined as the error between the calculated value and the best known accurate results and it is calculated as

$$\text{Mean Perchantage Error} = \frac{\text{Calculated value} - \text{Best Accurate value}}{\text{Best Accurate value}} \times 100 \quad (5.1)$$

Results involving the different methods are computed and compared with the benchmark test results. In the following tables, second column gives the instance to worked upon, third

column is the number customer nodes, fourth and fifth column records the benchmark results for the best accurate objective value and number of vehicles (NV), similarly sixth and seventh column records the calculated result and eighth column gives the Mean Perchantage error according to the equation (5.1).

Table 5.5. Comparison of Results (involving Cartesian method) with benchmark results

Sr. No	**Instance**	**N**	**Best Accurate known**		**Calculated using Cartesian Method**		**Mean Perchantage Error**
			Objective function	**NV**	**Objective function**	**NV**	
1	R101	25	617.1	8	618.341285	6	0.201
2	R101	50	1044	12	1047.12	11	0.298
3	C101	25	191.3	3	191.832	3	0.278
4	C101	50	362.4	5	363.2128	5	0.224

Table 5.6. Comparison of Results (involving hamming method) with benchmark results

Sr. No	**Instance**	**N**	**Best Accurate known**		**Calculated using Hamming Method**		**Mean Perchantage Error**
			Objective function	**NV**	**Objective function**	**NV**	
1	R101	25	617.1	8	618.34	6	0.2007
2	R101	50	1044	12	1046.854	11	0.2733
3	C101	25	191.3	3	191.831109	3	0.227
4	C101	50	362.4	5	363.0001	5	0.1655

Table 5.7. Comparison of Results (involving Brute-Curtis method) with benchmark results

Sr. No	**Instance**	**N**	**Best Accurate known**		**Calculated using Brute-Curtis Method**		**Mean Perchantage Error**
			Objective function	**NV**	**Objective function**	**NV**	
1	R101	25	617.1	8	617.2	6	0.0162
2	R101	50	1044	12	1044.23	11	0.02203
3	C101	25	191.3	3	191.38	3	0.0261
4	C101	50	362.4	5	362.523	5	0.03394

Table 5.8. Comparison of Results (involving Chebyshev method) with benchmark results

Sr. No	Instance	N	Best Accurate known		Calculated using Chebyshev Method		Mean Perchantage Error
			Objective function	NV	Objective function	NV	
1	R101	25	617.1	8	615.22	6	-0.304
2	R101	50	1044	12	1040.45	11	-0.340
3	C101	25	191.3	3	189.9	3	-0.731
4	C101	50	362.4	5	359.6	5	-0.772

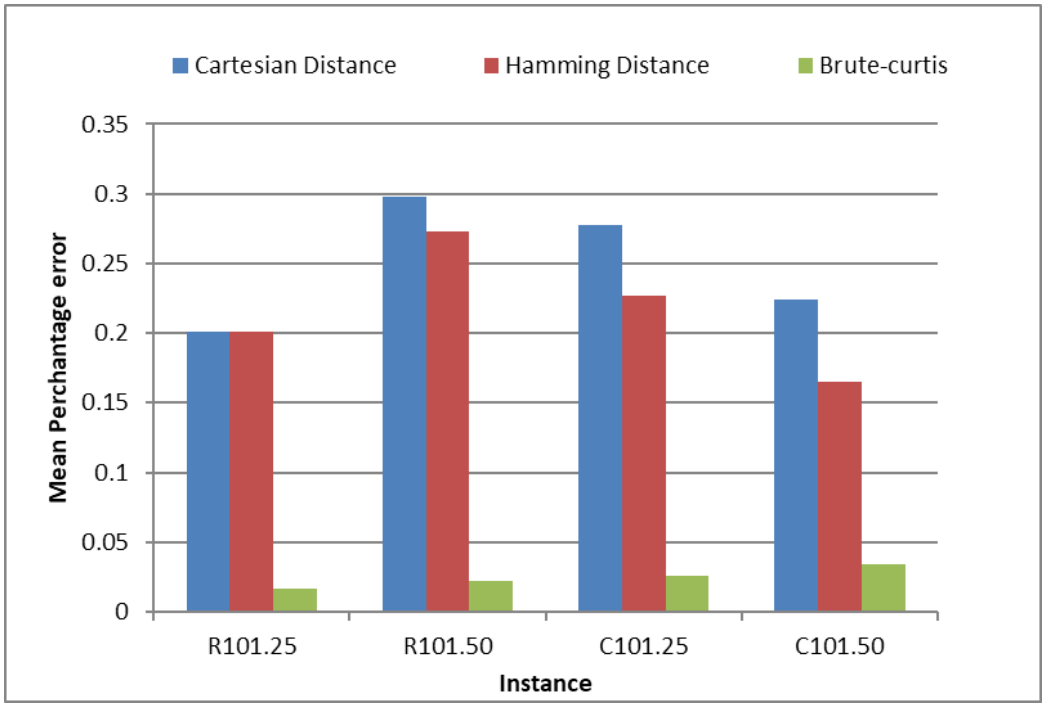


Figure 5.6. Comparison of mean perchantage error

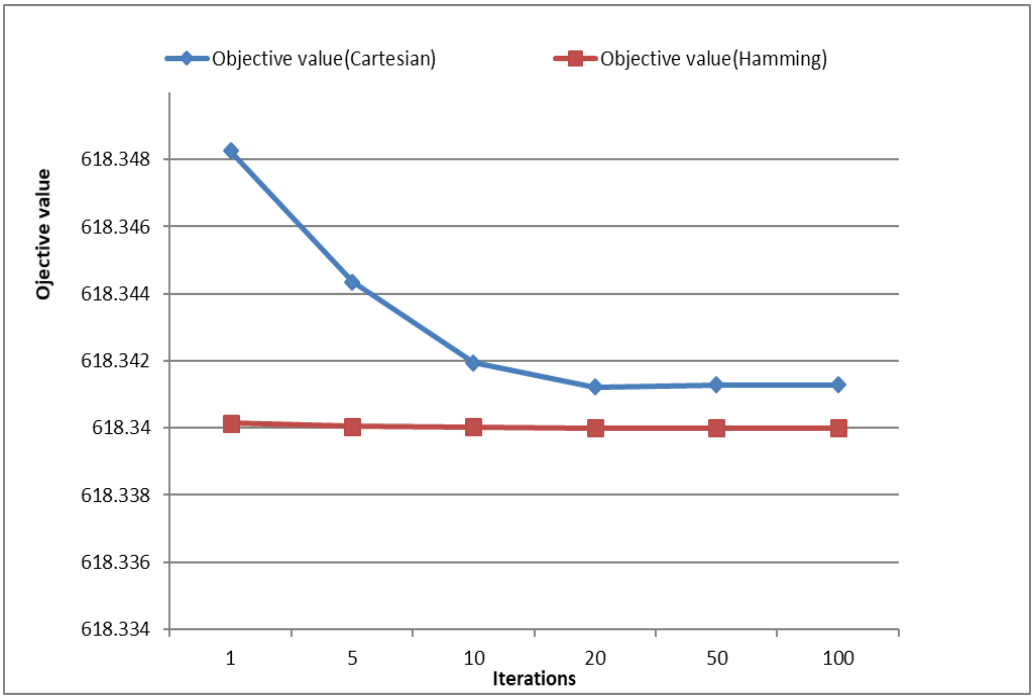


Figure 5.7. Comparison of Convergence Analysis of Objective value of R101.25

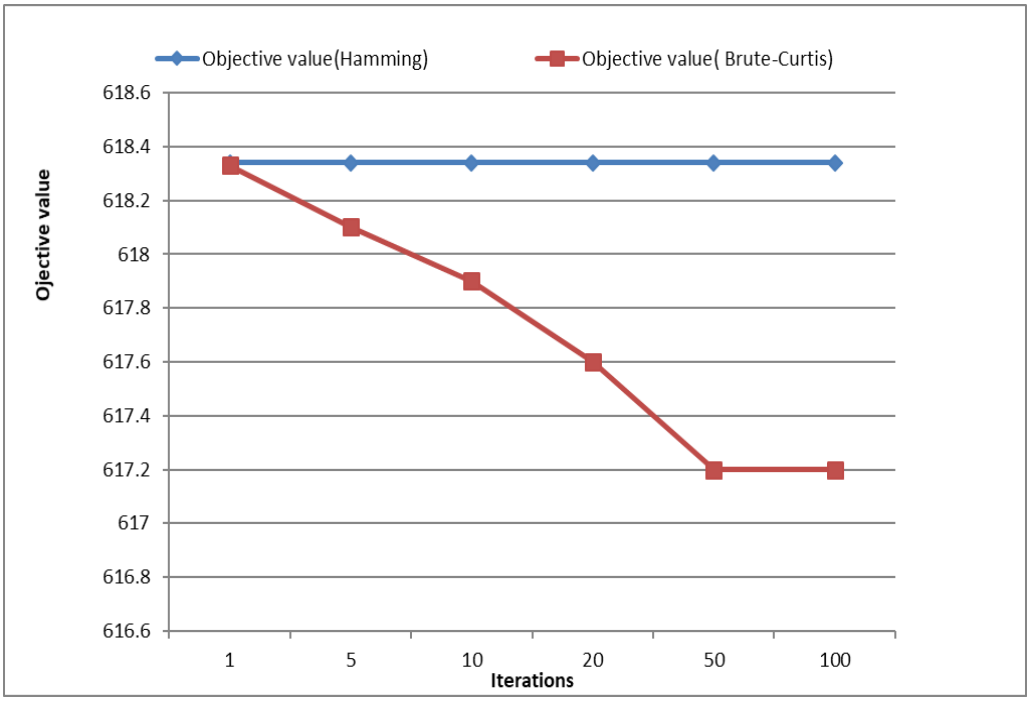


Figure 5.8. Comparison of Convergence Analysis of Objective value of R101.25

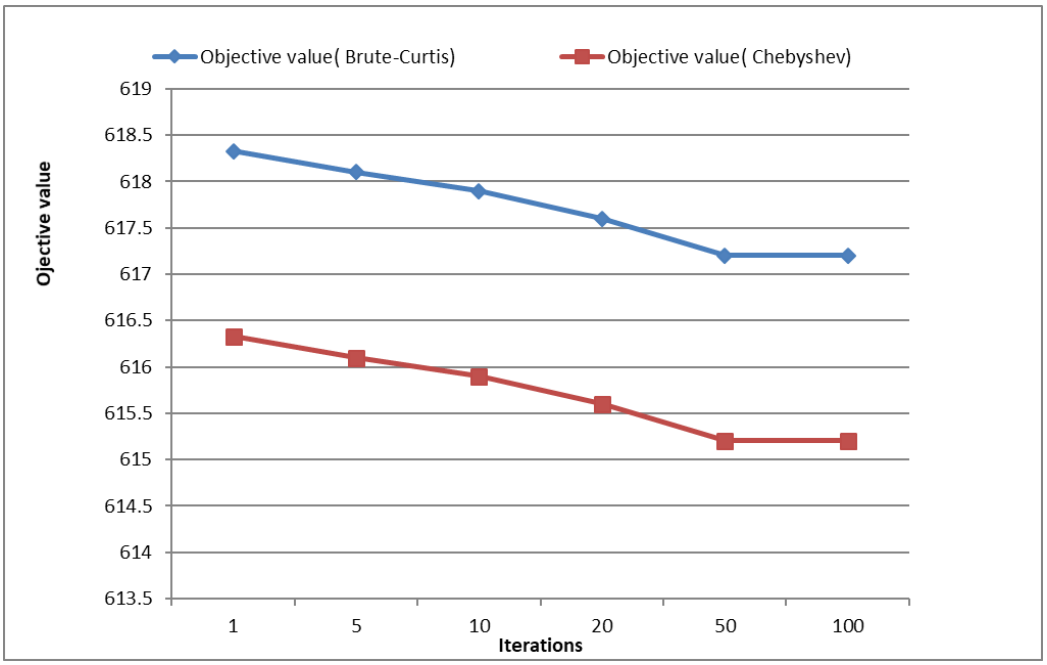


Figure 5.9. Comparison of Convergence Analysis of Objective value of R101.25

It can be observed from the tables that results computed with the Cartesian, Hamming and Brute-Curtis method are not as good as results computed from the Chebyshev method. Results computed from the Chebyshev method outperforms the benchmark results. Figure 5.6 shows the comparison of the Mean Perchantage errors computed from results of the former three methods. Brute-Curtis performs better than hamming and Cartesian method. Figure 5.7, 5.8 and 5.9 gives the convergence analysis of the objective function with the increase in iteration for the two consecutive distance metrics.

5.5. Improved Firefly Algorithm for VRPTW: Results and Discussion

The algorithm is implemented in CPLEX 12.6.3 on a PC clocked at 2.2 GHz under the operating system Windows 7

5.5.1. Test Instances

Here, two cases are discussed to test the Proposed Algorithm.

Case 1: To check the effectiveness of the algorithm, here we have used the testing example given by Wu *et al.* [59]. The proposed algorithm works on problem having 8 customers

and 3 vehicles and parameters from Table 5.8. and 5.9. with vehicle capacity of 8 tons. The results are computed and compared with that of the improved Particle Swarm Intelligence approach suggested in [41].

Table 5.9. Customer demand and time intervals

Index	1	2	3	4	5	6	7	8
Demand	2	1.5	4.5	3	1.5	4	2.5	3
S	1	2	1	3	2	2.5	3	0.8
$[e_i, l_i]$	[1,4]	[4,6]	[1,2]	[4,7]	[3,5.5]	[2,5]	[5,8]	[1.5,4]

Table 5.10. Distance between the nodes

D_i	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**0**
0	0	40	60	75	90	200	100	160	80	0
1	40	0	65	40	100	50	75	110	100	40
2	60	65	0	75	100	100	75	75	75	60
3	75	40	75	0	100	50	90	90	150	75
4	90	100	100	100	0	100	75	75	100	90
5	200	50	100	50	100	0	70	90	75	200
6	100	75	75	90	75	70	0	70	100	100
7	160	110	75	90	75	90	70	0	100	160
8	80	100	75	150	100	75	100	100	0	80
0	0	40	60	75	90	200	100	160	80	0

We evaluated the problem 100 times and analysed that the rate for algorithm computing the best solution is 93% which is greater than the 82% computed for improved PSO. However, the proposed approach follows the randomisation approach and reduces the problem space at each iteration, hence it is much faster than the improved PSO.

Routes for the best solution computed are 0-3-1-2-0, 0-6-4-0, 0-8-5-7-0

Case 2: The described Modification to the firefly Algorithm is here compared and analysed using the benchmark results obtained for Sololmon's (1987) well-known 56 benchmark problems [58]. These problem work on single depot, surrounding hundred customers, capacity constraint, service time constraint and time interval constraint. The whole instance are divided into three types of classes namely R, C and RC. The complete description about these instances is done in Appendix A.

5.5.2. Evaluated Results

Here 100 customer instances are considered and results are computed and compared with optimal solutions and best approximate solutions known. Table 5.10-5.12 records these results. Here the problem considers minimisation of total transportation cost and number of vehicles, so, both the parameters are compared.

Table 5.11. Computed results for the instances of class R

Instance	Optimal Solutions		Best Approximate Solutions		Our Solution	
	Distance	NV	Distance	NV	Distance	NV
R101	1637.7	20	1645.79	19	1638.3	18
R102	1466.6	18	1486.12	17	1473.8	17
R103	1208.7	14	1292.68	13	1244.2	12
R104	971.5	11	1007.24	9	989.4	9
R105	1355.3	15	1377.11	14	1363.2	13
R106	1238.6	13	1251.98	12	1245.8	12
R107	1064.6	11	1104.66	10	1178.0	10
R108	-	-	960.88	9	947.2	8
R109	1146.9	13	1194.73	11	1167.5	11
R110	1068	12	1118.59	10	1083.32	10
R111	1048.7	12	1096.72	10	1067.2	10
R112	-	-	982.14	9	973.6	8
R201	1143.2	8	1252.37	4	1240.65	4
R202	-	-	1191.70	3	1184.49	3
R203	-	-	939.54	3	936.0	3
R204	-	-	825.52	2	821.4	2
R205	-	-	994.42	3	985.8	3
R206	-	-	906.14	3	899.5	3
R207	-	-	893.33	2	886.4	2
R208	-	-	726.75	2	720.6	2

R209	-	-	909.16	3	904.8	3
R210	-	-	939.36	3	936.7	3
R211	-	-	892.71	2	887.42	2

Table 5.12. Computed results for the instances of class C

Instance	**Optimal Solutions**		**Best Approximate Solutions**		**Our Solutions**	
	Distance	**NV**	**Distance**	**NV**	**Distance**	**NV**
C101	827.3	10	828.94	10	828.9	10
C102	827.3	10	828.94	10	827.6	10
C103	826.3	10	828.94	10	827.8	10
C104	822.9	10	828.94	10	823.8	10
C105	827.3	10	828.94	10	827.42	10
C106	827.3	10	828.94	10	827.56	10
C107	827.3	10	828.94	10	827.6	10
C108	827.3	10	828.94	10	828.2	10
C109	827.3	10	828.94	10	827.3	10
C201	589.1	3	591.56	3	591.6	3
C202	589.1	3	591.56	3	590.8	3
C203	588.7	3	591.17	3	589.2	3
C204	588.1	3	590.60	3	588.34	3
C205	586.4	3	588.88	3	588.6	3
C206	586.0	3	588.49	3	588.4	3
C207	585.8	3	588.29	3	585.4	3
C208	585.8	3	588.32	3	586.2	3

Table 5.13. Computed results for the instances of class RC

Instance	Optimal Solutions		Best Approximate Solutions		Our Solution	
	Distance	NV	Distance	NV	Distance	NV
RC101	1619.8	15	1696.94	14	1656.87	13
RC102	1457.4	14	1554.75	12	1520.3	12
RC103	1258.0	11	1261.67	11	1260.3	11
RC104	-	-	1135.48	10	1133.2	9
RC105	1513.7	15	1629.44	13	1587.8	13
RC106	-	-	1424.73	11	1400.3	11
RC107	1207.8	12	1230.48	11	1225.4	11
RC108	1114.2	11	1139.82	10	1134.4	10
RC201	1261.8	9	1406.91	4	1345.2	4
RC202	1092.3	8	1367.09	3	1264.6	3
RC203	-	-	1049.62	3	1004.5	3
RC204	-	-	798.41	3	785.3	3
RC205	1154.0	7	1297.19	4	1110.4	4
RC206	-	-	1146.32	3	1150.2	3
RC207	-	-	1061.14	3	1084.3	3
RC208	-	-	828.14	3	827.9	3

5.4.3 Performance Evaluation:

The computational results show that the proposed approach give results better than the best approximate results in most of the cases. However, in some cases the results are not as good as best known results. The number of vehicles used is also optimised. Thus the results evaluated are encouraging and shows that the proposed algorithm is very promising.

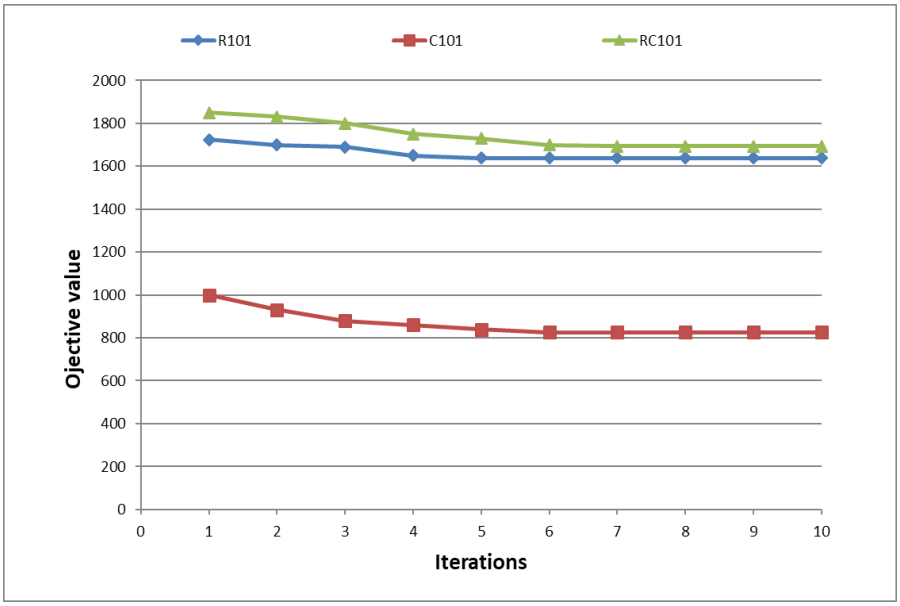


Figure 5.10. Convergence of first instance of each class

Compared to results in [60], the results are better and, moreover here the number of vehicle is also minimised. Table 5.11-5.13 shows that the proposed algorithm can achieve best of the solutions that can be evaluated to have a high convergence speed. The time taken by the computation of the results lies in range of several minutes for 50 iterations. Figure 5.12 shows the convergence rate of the first instances of each class namely- R101, C101, RC101.

Chapter 6
Conclusion and Future Work

6.1. Conclusion

The aim of this work is to design and develop an efficient solution for the most studied VRP variant Vehicle Routing Problem with time windows. Here, more than one solutions have been suggested efficiently. Firstly, mixed integer programming has been solved to minimize transportation cost and number of vehicles used simultaneously. It reveals that both the cost and number of vehicles are optimized at reasonable time. Secondly, a lagrangian based relaxation approach for the VRPTW has been suggested which relaxes the constraint set visiting all the customers. The instances are solved to 100 customer nodes. For larger instances, exact approaches becomes intensive in nature. Therefore, an approximation algorithm has been proposed to work on the VRPTW effectively. A modification to the Firefly algorithm has been proposed. Proposed distance calculation has converged the results to further extent. Moreover, reduction of problem space has lead the computation to become faster. The proposed approach has been tested on the well-known Solomon's benchmark test instances. Extensive results have shown that the proposed approach outperforms the benchmark results and is comparable to optimum results.

6.3. Scope for Future Work

The future directions for this work is describes as follows:

- The use of randomised approach for generation of initial population makes the problem inconsistent. Therefore, heuristics can be used to develop initial population.
- Metaheuristic can be integrated with Mixed Integer Programming to improve their performance.
- Investigation of different constraint for VRPTW under different environment would be a valuable contribution.

REFERENCES

[1] Dantzig, J. H. Ramser. “The Truck Dispatching problem”. Management Science, vol. 6, 1959, 80-92.

[2] M. Fisher and R. Jaikumar. “A Generalized Assignment Heuristic for Vehicle Routing”. Networks, vol. 11, 1981, pp. 109-124.

[3] P. Jaillet and M. R. Wagner. “Online vehicle routing problems: A survey”. In The Vehicle Routing Problem: Latest Advances and New challenges, B.L. Golden, S. Raghavan, and E.A. Wasil, eds., Operations Research/ Computer Science Interfaces Series, Springer, vol. 43, 2008, pp. 221-237.

[4] Haghani and S. Jung. “A dynamic vehicle routing with time dependent travel times”. Computer and Operations Research, vol. 32, 2005, pp. 2959-2986.

[5] J. Q. Li, P. B. Mirchandani and D. Borenstein. “Real time vehicle Routing Problem with time windows.” European Journal of Operation Research, vol. 194, 2009, pp. 711-727.

[6] J. F. Bard, G. Kontoravdis, G. Yu. “A branch-and-cut procedure for the vehicle routing problem time windows”. Transportation Science, vol. 36, 2002, pp. 250-269.

[7] J. F. Cordeau, G. Laporte and A. Mercier. “A unified tabu search heuristic for vehicle routing problem with time windows”. Journal of Operations Research Society, vol. 53, 2001, pp. 928-936.

[8] F. Maffioli. “The vehicle routing problem: A book review”. 4OR , vol. 1, 2003, pp. 149-153.

[9] G. Laporte. “The vehicle routing problem: An overview of exact and approximate algorithms”. European J. Oper. Res., vol. 59, 1992, pp. 345–358.

[10] P. Toth and D. Vigo. “An overview of vehicle routing problems”. 2001, pp. 1–26.

[11] M. L. Fisher. “Optimal solution of vehicle routing problems using minimum k-trees”. Operations Research, vol. 42(4), 1994, pp. 626–642.

[12] H. Pullen and M. Webb. “A computer application to a transport scheduling problem”. Computer Journal, vol. 10, 1967, pp.10–13.

[13] O. B. G. Madsen. “Optimal scheduling of trucks - A routing problem with tight due times for delivery”. In H. Strobel, R. Genser and M. Etschmaier. “Optimization applied to transportation systems”. IIASA, International Institute for Applied System Analysis, Laxenburgh, 1976, pp.126–136.

[14] K. Knight and J. Hofer. “Vehicle scheduling with timed and connected calls: A case study”. Operational Research Quarterly, vol. 19, 1968, pp. 299–310.

[15] B. L. Golden and A. A. Assad. “Perspectives on Vehicle Routing: Exciting New Developments”. Operations Research, vol. 34, 1986, pp. 803-809.

[16] B. L. Golden and A. A. Assad. “Vehicle Routing: Methods and Studies”. Elsevier Science Publishers, 1988.

[17] M. Desrochers, J. K. Lenstra, M.W.P Savelsbergh, and F. Soumis. “Vehicle Routing with Time Windows: Optimization and Approximation”. In: B. Golden and A. Assad (eds.). “Vehicle Routing: Methods and Studies”. Elsevier Science Publishers, 1988, pp. 65-84.

[18] A. W. J. Kolen, A. H. G. Rinnooy Kan, H. W. J. M. Trienkens. “Vehicle routing and scheduling with time windows”. Operations Research, vol. 35(2), 1987, pp. 266–273.

[19] R. T. Sam. “Vehicle Routing with Time Windows using Genetic Algorithms”: submitted to the book on Application Handbook of Genetic Algorithms, New Frontiers, 1995, pp. 253-277.

[20] R. Dondo, C. A. Mendez, J. Cerda. “An optimal approach to the multiple-depot heterogeneous vehicle routing problem with time window and capacity constraint”. INTEC (UNL-CONICET) Guemes, 3450-3000, Santa Fe –ARGRNTINA, 2003.

[21] R. Dondo, J. Cerda. “A cluster-based optimization approach for the multi-depot heterogeneous fleet vehicle routing problem with time windows”. European Journal of Operational Research, vol. 176, 2007, pp.1478–1507.

[22] C. Cetinkaya, I. Karaoglan, H. Gokcen. “Two-stage vehicle routing problem with arc time windows: A mixed integer programming formulation and a heuristic approach”, European Journal of Operational Research, vol. 230, 2013, pp. 539-550.

[23] S. Suwansuksamran, P. Ongkunaruk. “A Mixed Integer Programming For A Vehicle Routing Problem With Time Windows: A Case Study Of A Thai Seasoning Company”, Proceedings of the 4th International Conference on Engineering, Project, and Production Management, 2013.

[24] H. Simbolon, and H. Mawengkang. “Mixed Integer Programming Model For The Vehicle Routing Problem With Time Windows Considering Avoiding Route”. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), vol. 3, 2014.

[25] N. R. Sabar, X. J. Zhang, A. Song. “A Math-Hyper-Heuristic Approach for Large-Scale Vehicle Routing Problems with Time Windows”. in Evolutionary Computation (CEC), IEEE Congress, 2015, pp. 830-837.

[26] P. Hokama, F. K. Miyazawa, E. C. Xavier. “A branch-and-cut approach for the vehicle routing problem with loading constraints”. Expert Systems with Applications, vol. 47, 2016, pp. 1–13.

[27] G. Ozbaygin, O. E. Karasan, M. Savelsbergh, H. Yaman. “A branch-and-price algorithm for the vehicle routing problem with roaming delivery locations”. Transportation Research Part B, vol. 100, 2017, pp. 115–137.

[28] M. Desrochers, J. Desrochers and M. Soloman. “A New Optimization Algorithm for the Vehicle Routing Problem with Time Windows”. Opns. Res., vol. 40, 1992, pp. 342-354.

[29] K. Halse. “Modelling and Solving Complex Vehicle Routing Problems”. Ph.D. Dissertation No. 60, Institute of Mathematical Statistics and Operations Research, Technical University of Denmark, DK-2800 Lyngby, Denmark, 1992.

[30] N. Kohl and O. B. G. Madsen. “An Optimization Algorithm for the Vehicle Routing Problem with Time Windows based on Lagrangean Relaxation”. Operations Research, vol. 45, 1997, pp. 395 -406.

[31] C. Martinhon, A. Lucena, N. Maculan. “A relax and cut algorithm for the vehicle routing problem”. Technical Report RT-05/00, Universidade Federal Fluminense, Niterói, Brasil, 2000.

[32] B. Kallehauge, J. Larsen, O.B.G. Madsen. “Lagrangean duality applied to the vehicle routing with time window”. Computers & Operations Research, vol. 33, 2006, pp. 1464–1487.

[33] H. Karimi and A. Seifi. “Acceleration of lagrangian method for the Vehicle Routing Problem with time windows”. International Journal of Industrial Engineering & Production Research, vol. 23, 2012, pp. 309-315.

[34] H. D Holland. “Adaptation in Natural and Artificial Systems”. The University of Michigan Press, Ann Arbor, MI, 1975.

[35] N. El-Sherbeny. “Resolution of a Vehicle Routing Problem with Multiobjective Simulated Annealing Method”. Ph.D. Dissertation, Faculte Polytechnique de Mons, Belgique, 2001.

[36] O. Beatrice, J. R. Brian, H. Franklin. “Multi-objective genetic algorithms for vehicle routing problem with time windows”. Applied Intelligence, vol. 24 (1), 2006, pp. 17-30.

[37] W. Chiang, R. Russell. “Simulated annealing metaheuristics for the vehicle routing problem with time windows”. Annals of Operations Research 63, 1996, pp. 3–27.

[38] F. Glover. “Tabu search. Part 1”. ORSA, Journal on Computing, vol. 1 (3), 1989, pp. 190-206.

[39] M. Pirlot. “General local search methods”. European Journal of Operational, vol. 92, 1996, pp. 493–511.

[40] S. Amini, H. Javanshir and R. Tavakkoli-Moghaddam. “A PSO Approach for solving VRPTW with real case study”. IJRRAS, August 2010.

[41] Q. Zhu, L. Qian, Y. Li and S. Zhu, “An Improved Particle Swarm Optimization Algorithm for Vehicle Routin Problem with Time Windows”, 2006 IEEE Congress on Evolutionary Computation, Vancouver, BC, Canada, July 16-21, 2006.

[42] X.S. “Nature-inspired metaheuristic algorithms”. Luniver press, Bristol, 2008.

[43] X.S. Yang, “Metaheuristic optimization: algorithm analysis and open problems”. In: Experimental Algorithms, Springer, 2011, pp. 21–32.

[44] S. Das, S. Maity, B. Y. Qu, P.N. Suganthan. “Real-parameter evolutionary multimodal optimization survey of the state-of-the-art”. Swarm Evol. Comput., vol. 1(2), 2011, pp. 71–88.

[45] M. Sayadi, R. Ramezanian, N. Ghaffari-Nasab. “A discrete firefly meta-heuristic with local search for makespan minimization in permutation flow shop scheduling problems”. Int. J. Ind. Eng. Comput., vol. 1(1), 2010, pp. 1–10.

[46] O. Abedinia, N. Amjady, M.S. Naderi, “Multi-objective environmental/economic dispatchusing firefly technique”. In: IEEE International Conference on Environment and Electrical Engineering, 2012, pp. 461–466.

[47] K. Durkota. “Implementation of a discrete firefly algorithm for the qap problem within the sage framework”. BSc thesis, Czech Technical University, 2011.

[48] M.K. Marichelvam, T. Prabaharan, X.S.Yang. “A discrete firefly algorithm for the multiobjective hybrid flowshop scheduling problems”. EEE Trans., Evol. Comput. 18(2), 2014, pp. 301–305.

[49] Jati, G.K., et al.: Evolutionary discrete firefly algorithm for travelling salesman problem. In: Adaptive and Intelligent Systems (2011).

[50] S.L. Tilahun, H.C. Ong. "Modified firefly algorithm". J. Appl. Math., 2012,pp. 1-12.

[51] A. Gandomi, X.S. Yang, S. Talatahari, A. Alavi. "Firefly algorithm with chaos". Commun. Nonlinear Sci. Numer. Simul., vol. 18(1), 2013, pp. 89–98.

[52] L.D.S. Coelho, D.L. de Andrade Bernert, V.C. Mariani. "A chaotic firefly algorithm applied to reliability-redundancy optimization". In: IEEE Congress on Evolutionary Computation, IEEE, 2011, pp. 517–521.

[53] R. Aruchamy, K. Vasantha. "A comparative performance study on hybrid swarm model for micro array data". Int. J. Comput. Appl., vol. 30(6), 2011, pp. 10–14.

[54] T. Hassanzadeh, K. Faez, G. Seyfi. "A speech recognition system based on structure equivalent fuzzy neural network trained by firefly algorithm". In: IEEE International Conference on Biomedical Engineering, 2012, pp. 63–67.

[55] X.S. Yang. "Firefly algorithm, stochastic test functions and design optimisation". Bio-Inspired Computation, vol. 2, no. 2, 2010, pp. 78-84.

[56] X.S. Yang, S. DEB. "Eagle strategy using lévy walk and firefly algorithms for stochastic optimization". Studies in Computational Intelligence, vol. 284, 2010, pp. 101-111.

[57] V. Kumar, J. K. Chhabra, D Kumar. "Performance Evaluation of Distance Metrics in the Clustering Algorithms". INFOCOMP Journal of Computer Science, vol. 13(1), 2014, pp. 38-52.

[58] M. M. Solomon. "Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints". Operations Research, vol. 35, 1987, pp. 254-265.

[59] Y. Wu, C. M. Ye, H. M. Ma, M. Y. Xia. "Parallel particle swarm optimization algorithm for vehicle routing problems with time windows". Computer Engineering and Applications, vol. 43, no. 14, 2007, pp. 223-226.

[60] F. Pan, C. Ye, K. Wang, J. Cao. "Research on the Vehicle Routing Problem with Time Windows Using Firefly Algorithm". Journal of Computers, vol. 8, no. 9, sep. 2013.

Appendix A

Solomon Benchmark Test Instances

The Solomon's instances consists of 100 customers. There are three classes of the problem with two sets each. R, C, RC are the three classes. The name of the class symbolises the type of the customer in the class like, R class consists of randomly generated customer position, Class C consists of customers in clusters and Class RC consists of mixed type of customers locations. R1, R2, C1, C2, RC1 and RC2 are the six sets. The Perchantage of customers with time windows are 25, 50, 75 and 100% according to the problem respectively. R1, C1 and RC1 are the problem with short term planning horizon, low capacity vehicles and they allow only some customers (3-8) in each route. However, Sets R2, C2 and RC2 have long term planning horizon, high capacity vehicles, they are able to supply more than 10 customers per route. However, in each set of problem, the customer's position, demand and the service time do not change. So, problems R101 to R104 are identical, except they have different Perchantage of time windows for each one, 100% in instance R101, 75% in instance R102, 50% in instance R103 and 25% in instance R104. Instances from R105 to R108 are identical to problems from R101 to R104, the only difference is the time window interval. Similarly the R106 to R112 are same.

The complete description can be found at

https://www.sintef.no/projectweb/top/vrptw/solomon-benchmark/

List of Publications

Papers Published/ Accepted in International Journal

1. Mixed Integer Programming for the Vehicle Routing with Time windows, International Journal of Intelligent Systems Technologies and Applications. (Scopus Indexed)
2. A Comparative Analysis of Optimization Solvers, Journal of Information and Optimization Sciences. (ESCI)

Papers Published/ Accepted in International Conference

1. Lagrangian Relaxation for the Vehicle Routing Problem with time windows, IEEE International Conference on Intelligent Computing, Instrumentation & Control Technologies.

Papers Communicated in International Journal

1. Lagrangian Relaxation for the Vehicle Routing Problem with time windows, International Journal of Information Technology, Springer.

Video Link

https://youtu.be/nhUZwjsoifk