An Evolutionary Multi-Objective Approach for Fuzzy Vehicle Routing Problem

Zhavat Sherinov

Submitted to the
Institute of Graduate Studies and Research
in Partial Fulfilment of the Requirements for the Degree of

Master of Science
in
Computer Engineering

Eastern Mediterranean University
June 2011
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Elvan Yılmaz
Director

I certify that this thesis satisfies the requirements as a thesis for the degree of Master of Science in Computer Engineering.

Assoc. Prof. Dr. Muhammed Salamah
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Computer Engineering.

Asst. Prof. Dr. Ahmet ÜNVEREN
Supervisor

Examining Committee

1. Asst. Prof. Dr. Adnan ACAN
2. Asst. Prof. Dr. Ahmet ÜNVEREN
3. Asst. Prof. Dr. Önsen TOYGAR

ABSTRACT

In this thesis, Evolutionary Multi-objective Optimization Algorithm for solving Fuzzy Vehicle Routing Problem (FVRP) is described. FVRP is an extension of VRP with Time Windows, which is one of the variants of VRP. In addition to FVRP, Multiple Depot VRP (MDVRP) is used in solving the problem. So, the proposed work and the solution approach is a Fuzzy Multiple Depot VRP (FMDVRP). The objectives that are to be optimized in this solution approach are the minimization of: total travelled distance by vehicles, waiting time of vehicles and customers, and maximization of: load capacity of vehicles and service satisfaction of customers.

NSGA-II is a multi-objective optimization algorithm that is used for problems with several objectives to be optimized. In NSGA-II, there is population, which is initialized randomly, and then through several generations a new population is generated from the previous one, and the best of these populations are chosen. The typical genetic operators are applied for generating new population. In addition, NSGA-II uses a new parameter called crowding distance, which is used for better divergence.

In experimental results, benchmark problem instances classified by geographical distribution of customers are used in order to compare the results obtained with others. From the results, it is observed that the proposed solution minimizes the waiting time of vehicles by 30% more than the proposed solutions of other researchers.

Keywords: Fuzzy logic, multi-objective evolutionary algorithms, vehicle routing problems.

ÖZ

Bu tezde, Bulanık Araç Izgeleme Problemi (BAIP) için uygulanan Çok-Amaçlı Evrimsel Algoritması anlatılmıştır. BAIP zaman penceresi kısıtlamalı AIP'nin bir uzantısıdır ve AIP'nin bir çeşididir. BAIP'ye ek olarak, Çoklu Depo AIP (ÇDAIP) problemi çözmek için kullanılmıştır. Bu nedenle, önerilen çalışma ve çözüm yaklaşımı bir Bulanık Çok Depo AIP'dir (BÇDAIP). Bu çözüm yaklaşımı'nın amaçları: hedefler araçları'nın toplam katedilen mesafeyi, araçların ve müşterilerin toplam bekleme süresini en aza indirmek, araç yük kapasitesi ve müşteri hizmet memnuniyeti en yüksek seviyeye ulaştırmaktır.

Non-dominated Sorting Genetic Algorithm-II optimize edilmesi için çeşitli hedefler ile ilgili sorunlar için kullanılan çok amaçlı optimizasyon algoritmasıdır. NSGA-II de, başlangıçta nüfus başlatılır ve sonra birkaç nesil boyunca önceki oluşturulmuş nüfustan yeni bir nüfus oluşturulur ve bu toplumların en iyileri seçilir. Tipik genetik operatörler yeni nüfus üretmek için uygulanır. Buna ek olarak, NSGA-II, daha iyi sapma için, yeni kalabalık mesafe denilen bir parametre kullanır.

Deney sonuçlarında, başkaları ile elde edilen sonuçları karşılaştırmak için, müşterilerinin coğrafi dağılımına göre sınıflandırılan problem örnekleri kullanıldı. Sonuçlara göre, önerilen çözümün diğer araştırmacıların önerilen çözümlere göre 30% daha fazla araç bekleme süresini en aza indirdiği görülmüştür.

Anahtar Kelimeler: Bulanık Mantık, Çok-Amaçlı Evrimsel Algoritmalar, Araç Yönlendirme Sorunları.

ACKNOWLEDGEMENTS

First of all, I would like to say Alhamdulillah for giving me the strength and determination in completing this thesis throughout a year. I Praise Allah alone for His Mercy. I am very grateful to my parents for their invaluable support, patience and believe in me and who sacrificed everything for me.

My sincere thanks go to my supervisor Asst. Prof. Dr. Ahmet Ünveren for his assistance, direction and guidance. In particular, his recommendations and suggestions have been invaluable for the thesis, since he made me understand the topic about which I had little knowledge. I also wish to thank Asst. Prof. Dr. Adnan Acan for his shouldering while I was submitting a paper for publication related to my thesis.

I am particularly indebted to my wife for her invaluable patience, support and advices. I am thankful to her for restricting herself in many things during the time I was busy with thesis. I am also very grateful to my daughter Malika, who always used to increase my mood and make me happy.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

Throughout the history there were suggested many different variants of Evolutionary Algorithms (EA). EAs are heuristics which solve problems by using natural selection for searching. For the last several years, due to the significant development and improvement of algorithms EAs became very popular for searching and optimization tasks. The use of EAs is widespread in many fields for various problems – Vehicle Routing Problem (VRP), Travelling Salesman Problem (TSP), and Scheduling and other combinatorial optimization and integer programming problems.

A Genetic Algorithm (GA) is considered to be a subclass of EAs, which is a search heuristic that uses the method similar to the evolution process in nature. In GAs useful solutions or acceptable solutions are repeatedly generated over and over for optimization or search problems. However GAs are mainly applied to single-objective optimization problems unlike Multi-Objective Genetic Algorithms (MOGA) in which more than one objective functions are to be minimized or maximized concurrently. Therefore, in MOGA there cannot be only one solution, rather there are several solutions. So the aim is to search for all possible of the good solutions – called Pareto optimal set. Moreover, often, while solving real life problems the objective functions contradict to each other, since sometimes we need to minimize one objective and to maximize the other one.

In this thesis the Non-dominated Sorting Genetic Algorithm-II (NSGA-II) was used as the main search algorithm. NSGA II is a fast and elitist multi-objective genetic algorithm, which is an extension and improved version of genetic algorithm (NSGA), which also has dominancy, for multi-criteria optimization. However the difference between these algorithms is that in NSGA unlike NSGA II there is no elitism, also inefficient way of choosing a value for sharing parameter and it is not efficient in its computational complexity. Due to these criticisms of NSGA a new version was proposed by Kalyanmoy Deb et al. [1].

NSGA-II algorithm was used to search for the best solutions of Fuzzy Vehicle Routing Problem (FVRP). Vehicle Routing Problem (VRP) is a Non-Polynomial (NP) hard problem, where NP-hard problem is the problem whose solution is not verifiable in polynomial time. The main idea behind VRPs is that there is a single depot, a group of vehicles and a number of customers that should be serviced by these vehicles. One of the objectives is to minimize the travelled distance by all vehicles. However, there are other variants of VRP that are very effective in many application areas and also popular. One of those variants is a Multiple Depot VRP (MDVRP), which is also used in this thesis. MDVRP is very useful in problems where more than one depot exists, as in our case, since we have also different coordinates for vehicles other than the coordinates for customers.

In addition to MDVRP, VRP with Time Windows was applied in the proposed solution to the problem. This variant of VRP differs due to its time windows assigned to customers. These time windows are also called customers' preferences – their desired start time of service. VRPTW is very popular, since it is concerned with the customer preferences, therefore increasing the service satisfaction of customers. However, this approach is not enough to get full satisfaction of customers by using only VRPTW. Therefore, Fuzzy VRP was used as a better alternative to VRPTW, since FVRP gives more possibilities to vehicles in finding better and shorter routes within certain constraints than VRPTW. The difference, between FVRP and VRPTW is that in FVRP there exists so called fuzzy concept of *due-time*, which is very important in increasing service satisfaction of customers.

The experimental results were carried out using several benchmark problem instances – Solomon benchmark problems, which are classified by geographical distribution of customers. Also, the proposed work was experimented with these benchmark problem instances for several population sizes.

The organization of other chapters in this thesis is as follows. Chapter 2 presents the general description of VRP, MDVRP, VRPTW and Fuzzy VRP with their objectives and constraints. In Chapter 3, the genetic algorithms are explained with multi-objective optimization problems. Chapter 4 focuses on the proposed work and solution approach. The experimental results with different population sizes and for different combinations of objective functions are described in Chapter 5. Finally, the conclusion and discussion of possible future work is presented in Chapter 6.

Chapter 2

VEHICLE ROUTING PROBLEMS

2.1 Vehicle Routing Problem

Vehicle Routing Problem (VRP) is a well known integer programming problem and one of the combinatorial optimization tasks, which is in the category of NP hard problems. Initially it was defined more than 50 years ago by Dantzig et al [2] for solving a Track Dispatching Problem (TDP). In solving VRP the computational complexity and required effort depends on the problem size. Thus, the computational effort increases exponentially with the problem size. Usually for such problems the approximate solutions are found and various heuristic methods are used for searching and solving the problem. The interest of VRP lies in its practical relevance and its computational complexity. In VRPs the main aim is to construct appropriate routes for a group of vehicles, which should serve the number of customers. This difficult combinatorial problem is a combination of well-studied problems: Travelling Salesman Problem (TSP) and Bin Packaging Problem (BPP). VRP is used in a variety of application areas. Some of the practical applications of VRP are Tour Planning, Logistics, Deliveries of Goods to Department Stores, Picking up Students by School Buses, Newspaper, Laundry and Mail Distribution, Maintenance Inspection Tours and Scheduling Problems [33].

A typical VRP includes the following components: 1) a single depot, 2) a group of vehicles and 3) a number of customers with demands.

In every vehicle there is a limited capacity, but this capacity is same for all vehicles. Therefore a vehicle can accepts a limited number of customers. Each customer has its own demand, and this demand may vary from one customer to another. Each vehicle leaves a common depot to service customers and then returns back. While solving this problem we need to be careful about the customers not to be serviced more than once by the same vehicle and we should note that the demand of a customer does not change. In below Figure 2.1 the initial illustration of a VRP is given and the final possible graph or one of the solution to it is given in Figure 2.2.

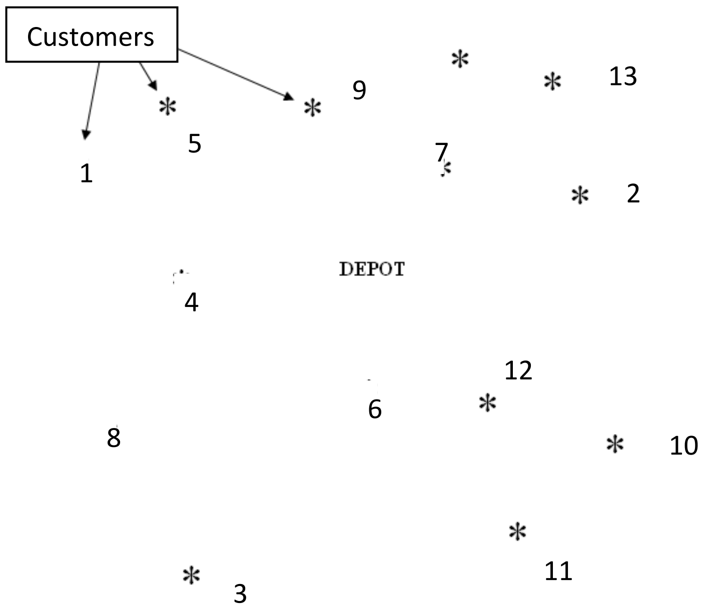

Figure 2.1. Sample input for Vehicle Routing Problem

$$R_1 = \{7,14,13,2\}$$
$$R_2 = \{4,1,5,9\}$$
$$R_3 = \{3,8\}$$
$$R_4 = \{12,10,11,6\}$$

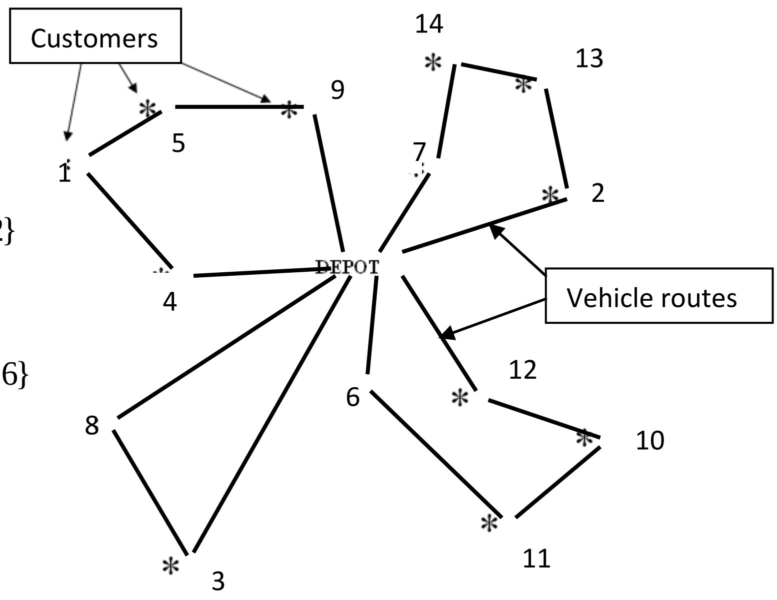

Figure 2.2. Possible solution for Vehicle Routing Problem

In illustrated figures above the customers are numbered from 1 to 14, and each vehicle has its own route or path. In above example we have 4 vehicles as shown next to the graph.

There are various variants of VRP that are used in a variety of application areas [34]. Some of them are:

- **Multiple Depot Vehicle Routing Problem (MDVRP)** – VRP with several depots.
- **Vehicle Routing Problem with Time Windows (VRPTW)** – VRP with the additional restriction that the time window is associated with each customer.
- **Capacitated Vehicle Routing Problem (CVRP)** – VRP with a constraint that every vehicle must have a uniform capacity.
- **Periodic Vehicle Routing Problem (PVRP)** – VRP which is generalized by extending the planning period to M days.
- **Split Delivery Vehicle Routing Problem (SDVRP)** – VRP in which the same customer can be served by different vehicles.
- **Stochastic Vehicle Routing Problem (SVRP)** – VRP in which one or several components of the problem are random.
- **Vehicle Routing Problem with Backhauls (VRPB)** – VRP in which customers can demand or return some commodities.
- **Vehicle Routing Problem with Pick-Up and Delivering (VRPPD)** – VRP in which the possibility that customers return some commodities is contemplated.
- **Vehicle Routing Problem with Satellite Facilities (VRPSF)** – VRP in which satellite facilities are used to replenish vehicles during a route.

In this thesis two variants of VRP were used for accomplishing the task – Multiple Depot VRP and VRP with Time Windows. Moreover, the Fuzzy VRP was applied, too. So we will focus on MDVRP, VRPTW and Fuzzy VRP.

2.2 Multiple Depot Vehicle Routing Problem (MDVRP)

In this thesis we have used a different approach with regard to a depot – Multiple Depot VRP (MDVRP). Unlike the description above, where we have a common depot from which vehicles start their path and then return back to the same depot again, we used a multi-depot approach. That means each vehicle has its own starting point, so for each vehicle there is a depot, whereas some vehicles may start from the same depot.

This variant of VRP – MDVRP is a very useful and popular in terms of application areas, since a firm might need or have more than one depot that are used by vehicles for serving customers. And it is known that MDVRP is NP-hard. The objectives of the problem are to service the customers while increasing their satisfaction grade and minimize the distance travelled by vehicles.

2.2.1 Related Work

In the literature there were done a lot of research and work by many researchers related to MDVRP. Some of the proposed works and solutions to different problems are mentioned below.

Canrong Zhang, Zhihai Zhang, Li Zheng and Linning Cai [3] proposed a multi-depot VRP with some customers, which do not yet belong to depots. Moreover, every depot has a limited number of vehicles. They constructed a mathematical model, which has two objective functions which are to minimize the total needed number of vehicles and the total route travelled.

Lei Wen and Fanhua Meng [4] improved Particle Swarm Optimization (PSO) for the Multiple Depot VRP with customer preferences – time windows. In their work, they also assume that a company may have several depots in which case VRP will not work. So in order to resolve the problem they focused on VRP with multiple depots and additionally with time windows (MDVRPTW).

Similar research was done by other researchers such as Michael Wasner et al. [5], where in their work they solved MDVRP by a proposed heuristic solution concept in which they used four time of feedback.

Wu TH et al. [6] proposed a different approach, in which they use two sub-problems by dividing multi-depot location-routing problem into two: location allocation and the general VRP.

Kaoru Hiorta et al. [7] used layers by decomposing the problem into three: Atomic layer, Molecular layer and Individual layer. In order to get better solutions they initially construct the routes, and the trip moves and exchange operations are performed between tours.

Wang Xuefeng [8] proposed a method to the integrated location-inventory-routing problem. The objectives of this problem are to minimize the total travel distance and, in addition, facility position cost, inventory cost and transportation cost.

Another work that was done using MDVRP variant is a research done by Richar Fung et al. [9], who proposed a Multiple Depot vehicle routing problem with weight-related cost (MDVRPWRC). In this extension they treat the vehicle load as a variable in the objective of model. In order to solve the MDVRPWRC they proposed a scatter search framework.

There are other extensions of MDVRP. For example, Salhi and Sari [10] proposed a variation of MDVRP, in which to minimize the total cost they considered a fleet with different vehicle capacities.

Dondo and Cerda [11] extended MDVRP to minimize the total cost including dispatching cost, distance and travel time cost, waiting time cost and penalty cost, by considering time-windows and heterogeneous vehicles together.

Proposed works and research mentioned above are examples of using MDVRP in various application areas and in addition to it the possible extensions of it. In the next section the classical MDVRP will be explained in details with an example.

2.2.2 Description and Formulation of MDVRP

As was mentioned in section 2.1, a company may have several depots in which case a MDVRP is used. The MDVRP needs to assign the customers to existing depots initially. Each of the depots has a fleet of vehicles assigned to it. Therefore, each vehicle leaves its depot and services the customers which belong to same depot, and then returns back to the original depot. The explanation is given below.

- **Objective:** The objectives are as follows: minimization of the travelled distance by vehicles and therefore the time of travelling, the total demand commodities of customers have to be serviced from different existing depots, and the number of vehicles needs to be minimized too.
- **Feasibility:** If normal constraints of VRP are satisfied and vehicles start servicing and end servicing at the same depot, and in that case the solution is accepted.

- **Formulation:** The cost of the route in MDVRP is calculated in the same way as in standard VRP. So, this is an extension of VRP in which we have more than one depot. And the set of vertices may be defined, i.e. the customers' positions as the following: $V = \{V_1,...,V_n\} \cup D$, where $D = \{D_1,...,D_m\}$ is the vertex, which represents the depots. So we have a route i as follows: $R_i = \{D_j, V_1,..., V_k, D_j\}$, where $D_j \in D$.

If the customers and depots are mixed in their positions, then MDVRP is solved. However, if the customers are grouped around depots, then we will need to have several independent VRPs. So the problem will be solved as a number of different VRPs. The sample input for the problem is given in Figure 2.3, and the possible solution to it is depicted in Figure 2.4.

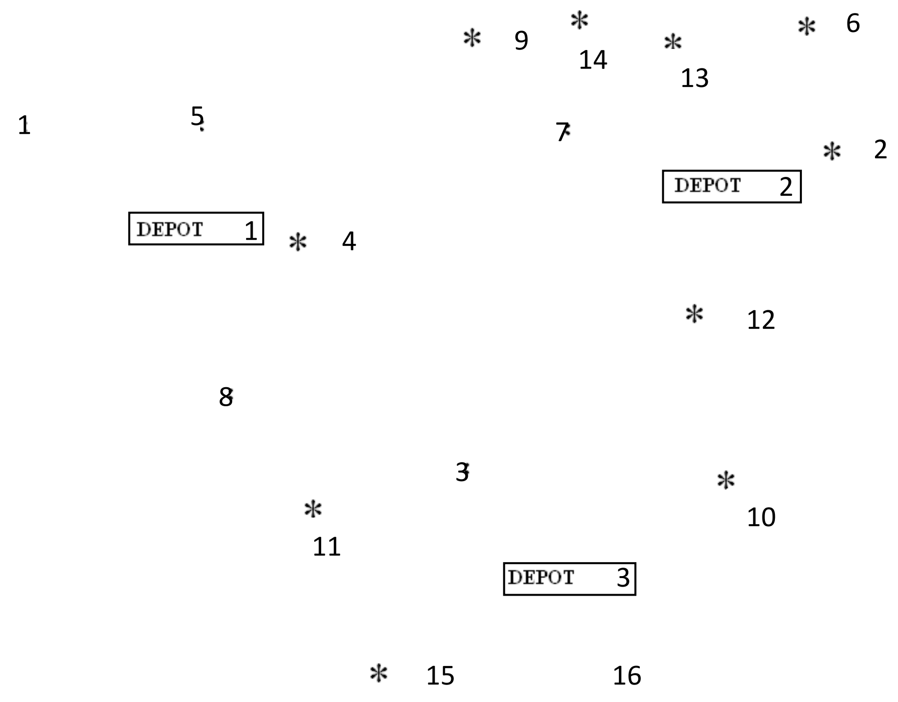

Figure 2.3. Sample input for MDVRP

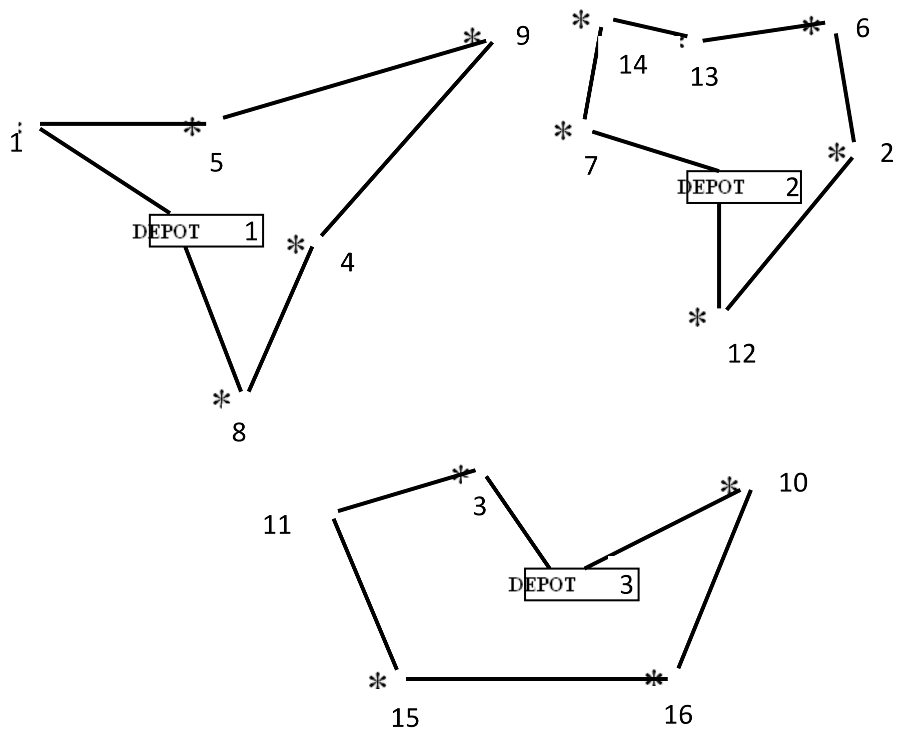

Figure 2.4. Possible solution for above MDVRP

In above Figure 2.3, there are three depots, and thus three vehicles, since in this example we assigned each vehicle to one depot. Each of the vehicles has its own path for servicing particular customers. The following are the routes for each vehicle belonging to different depot.

$$R_1 = \{D_1, 8, 4, 9, 5, 1, D_1\}$$
$$R_2 = \{D_2, 12, 2, 6, 13, 14, 7, D_2\}$$
$$R_3 = \{D_3, 3, 11, 15, 16, 10, D_3\}$$

This approach is much better in terms of finding good solutions, since there are more chances for vehicles to find appropriate customers with suitable demands and the distances to those customers.

2.3Vehicle Routing Problem with Time Windows (VRPTW)

Other than MDVRP, another variant of VRP was used in this thesis, which is an extension of it – Vehicle Routing Problem with Time Windows. In this variant of VRP there are certain time windows, which belong to each customer that should be served. In order to define this time window for a customer i, we use (e_i, l_i) denoting the earliest start of service time and the latest start of service time respectively for a customer to be served. This is the period of time during which customer must be serviced.

VRPTW is a very popular and has a high importance in heuristic, exact optimization algorithms and many application areas, since a lot of distribution systems include it. It is mainly concerned with the customer's demand by providing customers a service during the time they likes. And this increases the satisfaction grade of a customer.

2.3.1 Related Work

As was mentioned above, many researchers have made a lot of effort for both heuristic and exact optimization approaches related to VRPTW. Some of the research and works are mentioned below.

Between 1980's and 1990's first solution technique surveys were made by Golden and Assad [12] for the VRPTW. Also Desrochers et al. [13] and Solomon and Desrosiers [14] made a lot of effort in finding solution methods for VRPTW. Many heuristic approaches and exact methods were applied in order to solve VRPTW in history. As for the heuristic approaches some researchers did a significant rise in VRPTW in [15, 18, 20, 23, and 24]. Some of the recent researches related to VRPTW are mentioned below.

Olatz Arbelaitz et al. [16] managed a system, in which a Simulated Annealing is used in order to solve VRPTW problems. There are two optimization phases in the system, which was designed – a global one and the local one. Each of them is based on Simulated Annealing and parallelizable. The designed system could get the optimal solution, which is in most of the benchmark problems and, moreover, the averages of any group of random executions did not exceed five percent error than the best one.

Yiqing Zhong and Xiao Pan [17] proposed a hybrid solution in which they used simulated annealing algorithm in order to solve VRPTW. The hybrid optimization technique that they proposed includes the objective function and Simulated Annealing (SA) algorithm in order to deal with the VRPTW. The approach resulted in a good solution for the big size VRPTW and it is very efficient in terms of computational complexity with regards to standard SA algorithm.

Marco Antonio Cruz-Chávez et al. [19] proposed a Constraint Satisfaction Problem-VRPTW (CSP-VRPTW) algorithm. This algorithm applies the Precedence Constraint Posting (PCP) technique used to schedule as a Constraint Satisfaction Problem (CSP). The approach calculates the shortest path an both partial and global forms between all the nodes in the graph representing VRPTW model. This approach is efficient for some problems in the search of the global optimum.

Chengming Qi and Yunchuan Sun [21] proposed an Improved Ant Colony for VRPTW. So the algorithm is mainly based on Ant Colony System (ACS), which is hybridized with the randomized algorithm (RACS-VRPTW). In this work they had a couple of objectives – minimizing the set of routes or vehicles and minimization of total travel distance.

Chengming Qi et al. [22] proposed a solution to VRPTW by using Ant Colony System (ACS) and Local Search (LS) methods. The hybrid ACS that they proposed includes Randomized Algorithm (RA) and Pareto Local Search (PLS) algorithm (RPACS-VRPTW).The proposed algorithm can significantly reduce the costs of the constructive procedure of the solution in the ACS.

2.3.2 Description and Formulation of VRPTW

As was mention earlier the VRPTW is an extension of VRP and VRPTW is an NP-hard. What makes VRPTW different from VRP is that in VRPTW we have time windows associated with each customer. As illustrated below in Figure 2.5, the e_i is the earliest start of service time of customer i and the l_i is the latest start of service time of customer i to be served. An example of three customers with their preferences is depicted in Figure 2.6.

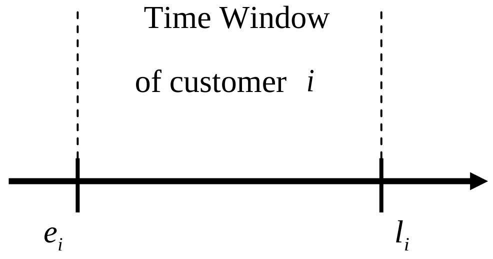

Figure 2.5. Time Window of customer i

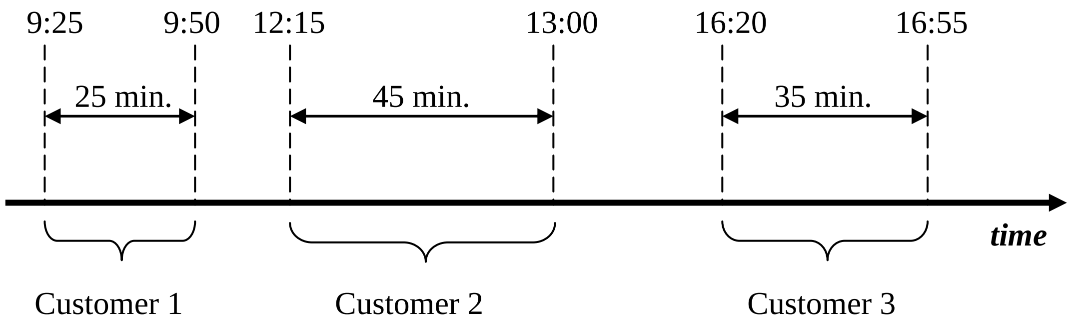

Figure 2.6. Time Windows for three customers

2.3.3 Formulation of VRPTW

The formal definition for VRPTW can be given as follows. There are V vehicles, k = 1, 2, 3… V. Each vehicle has its capacity C. We have N customers, i = 1, 2, 3 … N. Every customer has his/her own time window during which he/she likes to be served (e_i, l_i), where e_i denotes the earliest time of customer i and l_i denotes the latest time of customer *i*. The solution is infeasible, if vehicle starts servicing later than the latest time l_i and there occurs waiting time, if vehicle comes starts servicing before the earliest time e_i. A customer i has his/her service time denoted as t_i. Moreover each customer has a demand d_i. There exists a time needed to travel from customer i to customer j, and denoted as r_{ij}. The cost or distance from customer i to customer j is denoted by c_{ij}.

From above definition we derive *constants* and *variables* for the VRPTW as follows.

Constants:

V – is the number of vehicles.

C – is the capacity of a vehicle.

N – is the number of customers.

e_i – is the earliest time for customer *i*.

l_i – is the latest time for customer *i*.

c_{ij} – is the cost or distance from customer *i* to customer *j*.

d_i – is the demand of customer *i*.

Variables:

r_{ij} – is the required time to travel from customer *i* to customer *j*.

t_i – is the service time of customer *i*.

$$x_{ijk} = \begin{cases} 1, \text{ if vehicle } k \text{ travels directly from customer } i \text{ to customer } j \\ 0, \text{ otherwise} \end{cases}$$

After we defined constants and variables of VRPTW, we can formulate its *objective* and *constraints*.

Objective:

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

(2.1)

Subject to

Constraints:

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

(2.2)

$$\sum_{i \in N} d_i \sum_{j \in N} x_{ijk} \leq C, \qquad \forall k \in V$$

(2.3)

$$\sum_{j \in N} x_{ojk} = 1, \qquad \forall k \in V$$

(2.4)

$$\sum_{i \in N} x_{ijk} - \sum_{i \in N} x_{jik} = 0, \qquad \forall j \in N, \forall k \in V$$

(2.5)

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

(2.6)

$$x_{ijk}(t_i + r_{ij} - t_j) \leq 0, \qquad \forall i, j \in N, \forall k \in V$$

(2.7)

$$e_i \leq t_i \leq l_i, \qquad \forall i \in N, \forall k \in V$$

(2.8)

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

(2.9)

The objective function (2.1) is to minimize the total travel cost. Equation (2.2) ensures that each customer is visited by only one vehicle. Equation (2.3) ensures that a vehicle cannot exceed its capacity. Next, equation (2.4) and (2.6) denotes that a vehicle starts and ends traveling at the same depot. And equation (2.5) states that the distance from customer *i* to customer *j* and vice versa is equal. Equation (2.7) describes that the sum of service time of customer *i* and required time from customer *i* to customer *j* must be less than or equal to the service time of customer *j*. Next, equation (2.8) states the service time of customer *i* must be greater than or equal to the earliest time of customer *i* and less than or equal to the latest time of customer *i*, which means that the time windows are observed. Last equation (2.9) describes that variable x_{ijk} may have only 0 or 1.

2.4 Fuzzy Vehicle Routing Problem (FVRP)

Many problems are solved using VRPTW as it was mentioned in previous section. However, there are some real life problems that are not solved with VRPTW the way they should be, though VRPTW includes time windows, which increases the satisfaction grade of a customer. The lack of VRPTW is that it does not include the *due-time.* This approach is well developed in Fuzzy Logic – so called the concept of *fuzzy due-time.* This concept gives full flexibility and chance to a customer in order to increase his/her satisfaction grade as it is not restricted to earliest and latest times of a customer only as in VRPTW, but also there is penalty and reward for a vehicle servicing customers. Consequently Fuzzy *due-time* is responsible for satisfaction grade of a service time of customers. Therefore, it solves the problem in the best possible way and improves solution to the given problem [25].

2.4.1 Mathematical Model of Fuzzy VRP

Fuzzy vehicle routing problem is based on the concept of fuzzy *due-time*, where the membership fuzzy *due-time* corresponds to the satisfaction grade of a service time.

The customer's satisfaction for service is classified into two [25]:

- Tolerable interval of service time
- Desirable time for service

The tolerable interval of service time is shown in Figure 2.7. It is similar to VRPTW and has earliest service time e_i and latest service time l_i of customer ***i.*** However, the desirable time for service is denoted as u_i, which is a desired time as shown in Figure 2.8.

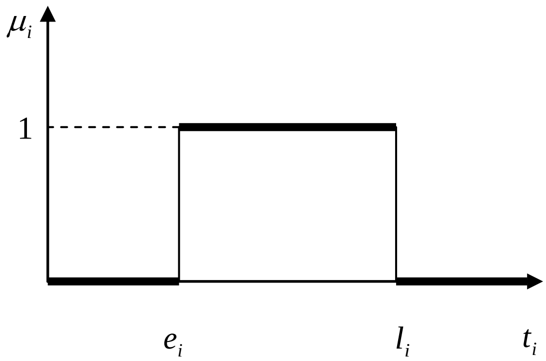

Figure 2.7. Tolerable interval for customer *i*

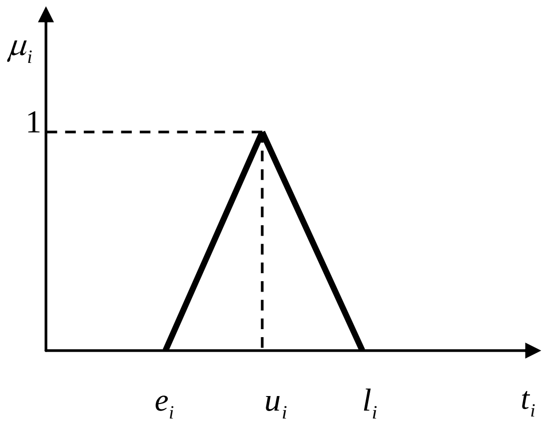

Figure 2.8. The desirable time for service

The desirable time for service is denoted as follows (e_i, u_i, l_i), where e_i and l_i are the earliest and latest times respectively as in VRPTW, and in addition to them there is an expected time or desirable time for customer *i*, which is u_i.

In Figure 2.9, the example of three customers is illustrated with different customers' preferences, since sometimes a customer may wish to be serviced earlier, so the tolerance of such customer will be less for earliness and vice versa.

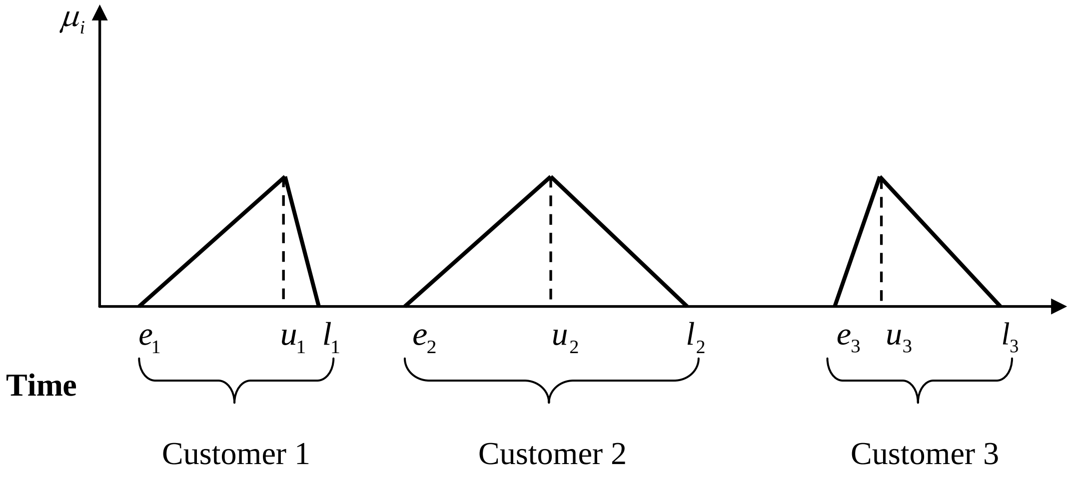

Figure 2.9. Fuzzy due-times for three customers

As shown in above figure, there are three customers with different service preferences. First customer's period between earliest service time and the desired time is big, but it is very small between the desired time and the latest time. Therefore, first customer is more tolerable for earliness and less tolerable for lateness, such that this customer cannot wait too much time. Third customer's preference is just opposite to the first one, since he/she is more tolerable for delay, but not for earliness. The second customer's preference as we see is equal for both earliness and lateness.

Constants:

V – is the number of vehicles.

C – is the capacity of a vehicle.

N – is the number of customers.

w_{kj} – is the waiting time for a vehicle *k* at customer *j*.

$\mu_i(t_i)$ – is the degree of satisfaction for customer *i*.

e_i – is the earliest time for customer *i*.

l_i – is the latest time for customer *i*.

u_i – is the expected or desired time for customer *i*.

c_{ij} – is the cost or distance from customer *i* to customer *j*.

d_i – is the demand of customer *i*.

Variables:

$$x_{ijk} = \begin{cases} 1, \text{ if vehicle } k \text{ travels directly from customer } i \text{ to customer } j \\ 0, \text{ otherwise} \end{cases}$$

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

The conventional models just consider the tolerance of customers as was mentioned above. This tolerance is represented as a time window and mathematical representation of it is as follows:

$$\mu_i(t) = \begin{cases} 1, \text{ if } e_i \le t \le l_i \\ 0, \text{ otherwise} \end{cases}$$

Where (e_i, l_i) represents the earliest and the latest start of service times of customer *i* and *t* is a time at which a customer is served.

However, in the desirable time for service a customer's preference is represented by a fuzzy approach as a triangular fuzzy number (TFN) and the membership function is defined as follows:

$$\mu_i(t_i) = \begin{cases} 0, & \text{if } t_i < e_i \\ \dfrac{(t_i - e_i)}{(u_i - e_i)}, & \text{if } e_i \le t_i \le u_i \\ \dfrac{(l_i - t_i)}{(l_i - u_i)}, & \text{if } u_i \le t_i \le l_i \\ 0, & \text{if } t_i > l_i \end{cases} \qquad (2.10)$$

Where (e_i, u_i, l_i) represents the earliest start of service time, desired start of service and latest start of service time of a customer *i*, and *t* is the time at which the customer is served. The above equation (2.10) will return zero if t_i is less than the earliest time of customer *i* or if t_i is greater than the latest time of customer *i*.

Otherwise, the equation will return a corresponding value calculating values of e_i, u_i, l_i with respect to t_i.

Objectives:

- Minimizing the fleet size.
- Minimizing the total travel cost.
- Maximizing the average satisfaction grade of customers.
- Minimizing the total waiting time for all vehicles.

Mathematically representation of objectives is as follows:

$$\max \frac{1}{n}\sum_{i\in N}\mu_i(t_i),$$

(2.11)

$$\min \sum_{j\in N}\sum_{k\in V} x_{0jk} ,$$

(2.12)

$$\min \sum_{i\in N} w_i(t_i),$$

(2.13)

Additional Equations and Constraints:

$$\mu_i(t_i) > 0, \qquad \forall i \in N$$

(2.14)

$$t_i \geq 0, \qquad \forall i \in N$$

(2.15)

$$w_k = e_j - t_j, \qquad \text{if } t_j < e_j$$

(2.16)

Where

$$t_j = t_i + r_{ij} + s_i$$

The objective (2.11) is for maximizing the average satisfaction grade for all customers. Next, the objective (2.12) is to minimize the vehicle fleet size. The objective (2.13) is used for minimizing the average waiting time of all customers.

The constraint (2.14) is to ensure that the service satisfaction grade is not zero or less. Next, the constraint (2.15) is to ensure that all customers' service times are non-negative. The equation (2.16) with its constraint is for calculating the waiting time, where t_j is the start of service time of customer *j*, t_i is the start of service time of customer *i*, r_{ij} is the time to travel from customer *i* to customer *j*, s_i is the service time of customer *i*.

2.4.2 Related Work

Fuzzy logic is applied in many application areas and in the area of artificial intelligence as well. Fuzzy logic enables people to get the flexible operations over some problems and makes it easy to get desired and better solutions.

There are some related works and researches mentioned below, which are done recently.

Jeng-Jong Lin [26] proposed a multi-criteria decision making for VRP with Fuzzy due time. He applied the fuzzy due time concept in order to change the standard VRPTW so that the customers' preferences and demand are met much better. In his Fuzzy vehicle routing five objectives are solved: service satisfaction, space-utility, load-focusing degree, waiting time and total distance travelled by vehicles. In this work he used a pure genetic algorithm to acquire the results for the problem.

Gupta et al. [27] proposed an extension to the fuzzy problem with multi-riteria optimization considered by Gupta et al. [28]. They added a new attribute, which is minimizing the number of vehicles. Moreover, they consider the time to be dynamic, since in practice the exact time taken to serve a customer is usually unpredictable.

In another work Gupta et al. [29] proposed an extension to [27], in which they consider a case study of Jain University of Bangalore in Karnataka. This university provides bus services to pick up and drop students and staff from/to home.

Chapter 3

MULTI-OBJECTIVE OPTIMIZATION PROBLEM

3.1 Genetic Algorithms

A genetic algorithm (GA) is a search heuristic that uses the method of natural evolution process. In GAs useful solutions or acceptable solutions are repeatedly generated over and over for optimization or search problems. However, GAs are mainly applied to single-objective optimization problems. So in GAs, there is a single objective that should be optimized depending on the given problem. The typical GA consists of the following components:

- **Individual** – any possible solution (also known as *chromosome*).
- **Population** – a set of individuals.
- **Parent** – member of a current generation.
- **Offspring** – generated child (individual) from parent(s), a member of next generation.
- **Generation** – iteratively created populations.

The classical GA works as follows: first, a population is created with a set of individual generated randomly. Next, these individuals are evaluated according to the objective function, which depends on the problem. After calculating the objective function values for each chromosome, the selection is applied based on their fitness to get parents for reproduction. If the fitness is high, then there are more chances for an individual to be selected. Then, genetic operators are applied on these parents (selected individuals) in order to generate new offspring (new population). The flowchart of typical GA is illustrated in Figure 3.1.

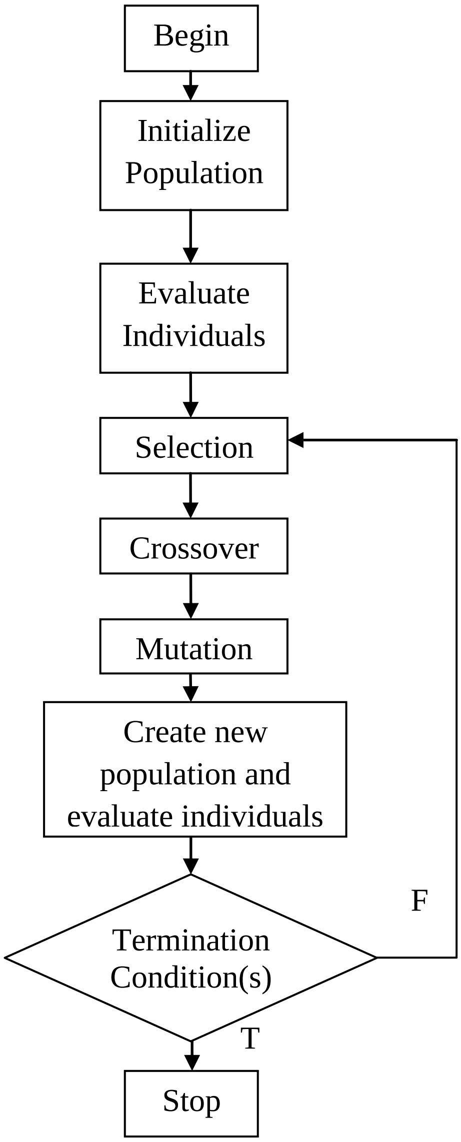

Figure 3.1. Genetic Algorithm Flowchart

As shown in the figure there is crossover and mutation. So they are the main operators that are applied on selected parents in order to generate offspring. If crossover is applied, then two offspring are generated, otherwise only one offspring is generated as a result of mutation. Since selection, crossover and mutation play a significant role in GAs, below a detailed description of them is given and different methods for each of them are illustrated.

3.1.1 Genetic Operators

3.1.1.1 Selection

Selection in GAs is very important, since it chooses the individuals (parents) from the population for generating next population. Therefore, these parents must be good enough to get a better population than the current one. Several approaches for selection methods are explained below.

3.1.1.1.1 Tournament Selection

In Tournament Selection we have a parameter *k*. Then we select two individuals and generate a random number between 0 and 1 inclusive. So, if the generated number is less than parameter *k*, then the best among the two selected individuals is chosen, otherwise we choose the worst one. The pseudo code for tournament selection is given below.

Tournament Selection pseudo code:

1. Select two individuals
2. Generate a random number *n*, $0 \le n \le 1$
3. Select the best of the 2 individuals, if $n<k$
4. Otherwise, select the worst of the 2 individuals

Where *k* is a parameter.

There are several benefits of tournament selection: it is easy to code, lets the selection pressure to be easily adjusted and works on parallel architectures.

3.1.1.1.2 Roulette Wheel Selection

Roulette wheel selection (RWS), also known as fitness proportionate selection (FPS), is another genetic operator used for selecting useful individuals for generating offspring. In RWS the fitness value is used to compare the probability of selection with each individual. So the probability of a chromosome (individual) being selected is defined as follows:

$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j}$, where f_i is the fitness of a chromosome i in the population and N is the size of population – the number of individuals in the population.

The pseudo code for RWS is given below:

RWS_pseudo_code(f)

1. Fitness sum = sum of all chromosomes' fitness's;
2. R = random value between 1 and Fitness sum;
3. Initialize sum with zero;
4. for *i*=1 to N
5.   sum = sum + fitness value of chromosome *i*;
6.   if sum >= R
7.   break;
8.   end
9. end
10. Parent = chromosome *i*;

3.1.1.2 Crossover

Crossover is also a genetic operator, which is used to combine two different chromosomes and produce two new chromosomes. And the idea here is to produce the better chromosomes than the previous two by taking the best characteristics of them. Mainly, the crossover is applied according to its probability set by the programmer. The following are some of the simple crossover types with their illustrations.

3.1.1.2.1 One-Point Crossover

One-point crossover is useful and easy to apply in GAs and it is simple to implement it in programming. This type of crossover works as follows: first a point is generated randomly between 1 and the length of the chromosome, and then the first part of parent 1 is concatenated with the second part of parent 2 and given to child 1. Also for child 2 the first part of parent 2 is concatenated with the second part of parent 1. In Figure 3.2, the illustration of one-point crossover is given.

Point

Parent 1: **101101101101010**|111010011011001

Parent 2: 001110110100011|**001011110110111**

Child 1: **101101101101010001011110110111**

Child 2: 001110110100011111010011011001

Figure 3.2. One-point crossover

3.1.1.2.2 Two-Point Crossover

Two-point crossover is similar to one-point crossover except that there are two points and the procedure is the same. The subparts of parents are swapped between offspring. The illustration of two-point crossover is depicted in Figure 3.3.

Point 1 **Point 2**

Parent 1: **10110110**|11010101110100|**11011001**

Parent 2: 00111011|**01000110010111**|10110111

Child 1: **101101100100011001011111011001**

Child 2: 001110111101010111010010110111

Figure 3.3. Two-point crossover

3.1.1.2.2 Uniform Crossover

Another type of crossover is a uniform crossover, in which there is a *mask* consisting of bits (zeros and ones) generated randomly. The uniform crossover distributes the bits according to the mask. So, if the bit in a *mask* is 1, then the bit from parent 1 is copied to child 1, else it is copied to the exact position as in parent 1 to child 2. The same procedure is applied for parent 2. Uniform crossover is illustrated in Figure 3.4.

***Mask* :** **011001010110011**

Parent 1: 1**00**11**1**1**0**0**10**10**10** **Parent 2:** **0**10**10**0**1**1**0**11**00**01

Child 1: **000101100100010** **Child 2:** 110110110111001

Figure 3.4. Uniform Crossover

3.1.1.3 Mutation

The mutation operator is a genetic operator that is responsible for changing the chromosome within itself by modifying its contents. A better solution, i.e. chromosome is expected to be gained by applying the mutation on it. Generally, mutation is applied on a single chromosome, but not on a pair of chromosomes as it is done in crossover operator. Some of the simple mutation types are illustrated below.

3.1.1.3.1 Point Mutation

In point mutation a particular point in a chromosome is selected and changed by some other value, which results in a different chromosome, the modified one. Illustration of point mutation is depicted in Figure 3.5.

Before: 011001**0**10110011
↓
After: 011001**1**10110011

Bit representation

Before: 2 3 1 **4** 7 6 5
↓
After: 2 3 1 **8** 7 6 5

Integer representation

Figure 3.5. Point Mutation

3.1.1.3.2 Swap Mutation

Swap mutation is similar to point mutation in a way that it also chooses a point, whereas it chooses two points and then swaps them with each other. Swap mutation is depicted in Figure 3.6.

Before: 01**1**0010101 1**0**011

After: 01**0**0011101 1**1**011

Bit representation

Before: 2 **3** 1 4 7 **6** 5

After: 2 **6** 1 4 7 **3** 5

Integer representation

Figure 3.6. Swap Mutation

3.1.1.3.3 Rotate Mutation

Rotate mutation is very useful technique for mutating a chromosome and is considered as a better approach than the previous two. The reason is that in rotate mutation there is no modification of genes themselves, and therefore we keep the genes the same but we just reorder them. This makes the chromosome to be different and modified. However, keeping genes same is very important in many application areas and problems, in which we cannot have a repeated gene. For example, in the Travelling Salesman Problem (TSP) each city is visited only once, so for such problem we will not have doubled cities in a chromosome, if we use rotate mutation. The rotate mutation works as follows: first a point is selected as in other types of mutation, and then chromosome is rotated depending on this point.

The illustration of rotate mutation is given in Figure 3.7.

Before: **01100101011**0|011

After: 0011**01100101011**

Bit representation

Before: **2 3 1 4 7 6** 5|8 9

After: 5 8 9 **2 6 1 4 7 3**

Integer representation

Figure 3.7. Rotate Mutation

3.1.1.3.4 Reordering

A good alternative to rotate mutation is reordering. This mutation is also very effective and in most of the problems gives good results from mutation operator. In reordering, first two points are selected, and then all elements or genes are reversed. The example of reordering technique is illustrated in Figure 3.8.

Before: 0110|**0101011**|0011
After: 0110|**1101010**|0011
Bit representation

Before: 2 3|**1 4 7 6**|5 8 9
After: 2 3|**6 7 4 1**|5 8 9
Integer representation

Figure 3.8. Reordering

3.2 Multi-Objective Genetic Algorithms

Multi-objective optimization is the process that optimizes several objectives concurrently, and these objectives are subject to some constraints. Multi-objective optimization is also called multi-criteria or multi-attribute optimization. Moreover, in multi-objective optimization problems (MOP), often, while solving real life problems the objective functions are conflicting and thus contradict to each other, since sometimes we need to minimize one objective function and to maximize the other one. Therefore, there is no single unique solution in MOP, rather there are several solutions. So the aim is to find all of the good trade-off solutions available – called Pareto optimal set. The following are the examples of real life problems with conflicting objectives, for which a MOP is solved: automobile design, finance, aircraft design or any other area in which the optimal decisions must be taken. For example, minimizing the cost of a product and maximizing the profit, maximizing the satisfaction grade of customers and minimizing the waiting time of vehicles or minimizing the distance travelled by a vehicle and maximizing the load capacity are all the examples of multi-objective optimization problems.

3.2.1 Formulation of Multi-objective Optimization

As was mention above, in MOPs there are several objectives that must be minimized or maximized instead of one objective as in GAs. In addition to it, there are constraints that these objectives are subject to. Below, the mathematical definition for multi-objective problem is given:

$$(\max\ or)\min[\ f_1(x),\ f_2(x),...,\ f_n(x)] \tag{3.1}$$

Subject to k inequality constraints:

$$g_i(x) \le 0 \qquad i = 1, 2, \ldots, k \tag{3.2}$$

And m equality constraints:

$$h_i(x) = 0 \qquad i = 1, 2, \ldots, m \tag{3.3}$$

Where n is the number of objective functions and x is the vector of decision variables, which is defined as follows: $x = [x_1, x_2, ..., x_n,]^T$. So, with particular set of values $- x*_1, x*_2, ..., x^*{}_n$, we aim to find the optimal solutions by minimizing (or maximizing) objective functions in (3.1) and at the same time satisfying constraints in (3.2) and (3.3).

3.2.2 Pareto Dominance in Multi-objective Optimization

The main difference between the genetic algorithms and the multi-objective optimization problems or multi-objective genetic algorithms (MOGA) is that, as was stated above, the MOPs optimize several objective functions concurrently, whereas in classical GAs only one objective function is optimized. So, the question is that how MOP solves the problems with multiple objectives that must be optimized. The solution for this is the *dominancy* in MOP. The dominancy solves the problem of multiple objectives and the idea behind of dominancy is as follows: the chromosomes, which are *non-dominated,* are in the optimal set of a solution. Therefore, they are chosen as the best found solutions for the given problem, if they are not dominated by any other individual in the population. And the dominancy rule is as follows:

*A state **A** dominates a state **B**, if **A** is better than **B** in at least one objective function and not worse with respect to all other objective functions.*

Mathematically, *Pareto dominance* is described as follows:

A vector $\vec{u} = (u_1, ..., u_k)$ is said to dominate $\vec{v} = (v_1, ..., v_k)$ (denoted by $\vec{u} \preceq \vec{v}$) if and only if *u* is partially less than *v*, such that $\forall i \in \{1, ..., k\}, u_i \leq v_i \wedge \exists i \in \{1, ...k\} : u_i < v_i$.

The illustration of Pareto dominance is depicted in Figure 3.9. As shown in this figure, for minimization problem, i.e. if we are minimizing two objective functions, then the dominated area is that which is on the upper right side of a particular solution. And the opposite for the maximization problems, the dominated area is on the left bottom side of a particular solution.

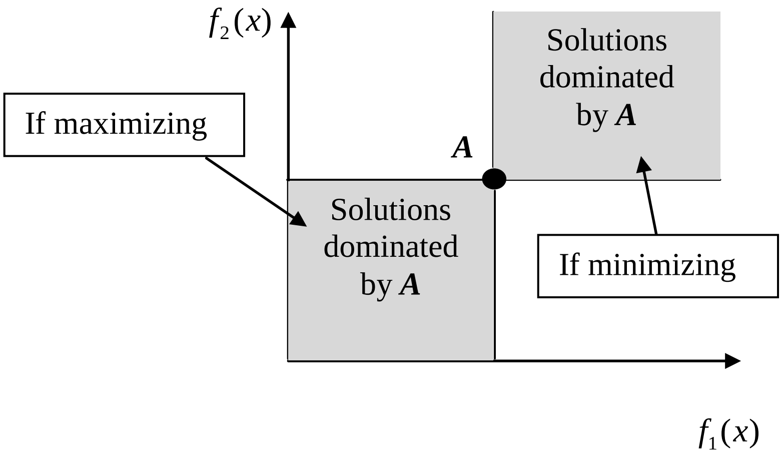

Figure 3.9. Dominated area for Pareto dominance

In Figure 3.10, two objectives with two individuals are represented to illustrate the Pareto dominancy.

Maximize $f(x) = (f_1(x), f_2(x))$

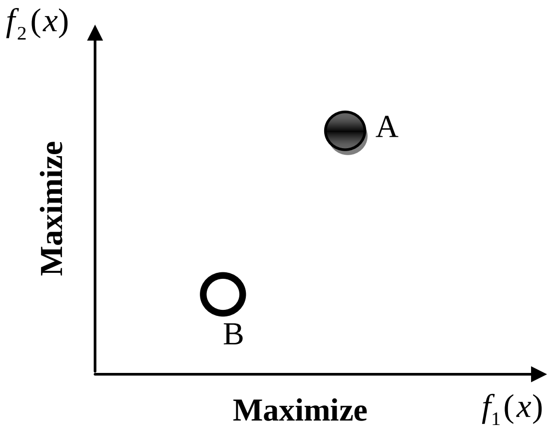

Figure 3.10. Pareto dominance

Considering the definition of Pareto dominance we can say the following about the above figure: **A** dominates **B**, so **B** is dominated by **A**, and since **A** is non-dominated it is better than **B**. In Figure 3.11, the Pareto optimal solutions are illustrated for minimization problem.

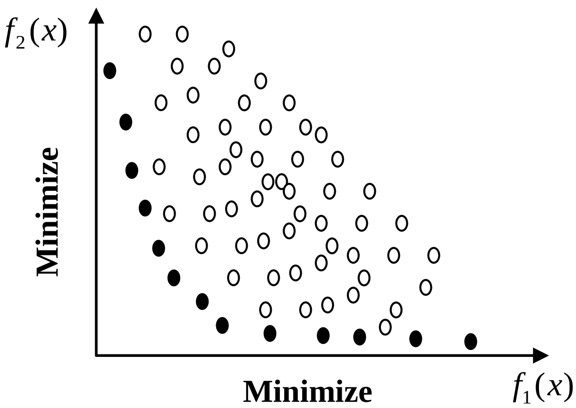

Figure 3.11. Pareto optimal solutions

The solutions that are colored black are the Pareto optimal solutions, such that they are non-dominated solutions for this minimization problem. All other solutions are dominated by these Pareto optimal solutions.

Chapter 4

SOLUTION WITH NON-DOMINATED SORTING GENETIC ALGORITHM-II FOR FUZZY VEHICLE ROUTING PROBLEM

In this chapter, the proposed work with sorting algorithm will be discussed, since the general idea of genetic algorithms and their multi-objective versions are presented in previous chapters along with various crossover and mutation techniques and the related work in literature. This chapter presents Non-dominated Sorting Algorithm-II (NSGA-II) [1] for Fuzzy Vehicle Routing Problem and its detailed description. NSGA-II is a very effective sorting algorithm, which can be applied in many application areas. It is an extension of Non-dominated Sorting Genetic Algorithm-I (NSGA-I), which is also a popular non-domination based genetic algorithm for multi-objective optimization and it is very effective. However, due to its computational complexity, the choice of the optimal parameter value for sharing parameter σ_{share} and the lack of elitism, it has been criticized. In NSGA-II, there is elitism and in addition to it no sharing parameter is needed to be chosen in advance. Therefore, it is a better sorting algorithm with comparison to NSGA-I. Moreover, in NSGA-II there is a *crowding distance* parameter used for ensuring the divergence in finding the Pareto optimal solutions, since it is a measure of how close an individual is to its neighbors. Therefore, the idea behind crowding distance is to have it large, which will give better solutions with diversity.

In this thesis, several objective functions were optimized while solving FVRP with NSGA-II. The objective functions implemented are the following: minimizing the distance travelled by vehicles, minimizing the waiting time of vehicles, maximizing the load capacity for vehicles, maximizing the satisfaction grade of customers and minimizing the waiting time of customers. However, our main concentration was on three objective functions, which are minimizing the waiting time of vehicles, minimizing the waiting time of customers, which we also call the delay times of vehicles and maximizing the satisfaction grade of customers. Therefore, these three objective functions will be discussed in detail, though in the experimental results, which is chapter 5, other objective functions' results also will be given.

4.1 General Framework of Non-dominated Sorting Genetic Algorithm-II

The following steps are made in NSGA-II. The algorithm first initializes the population and after that the population is sorted into each *front* based on non-domination. The first front is non-dominated one, the second front is dominated only by the first front, the third front is dominated by first and second and it goes like that. All fronts are numbered from 1 up to the total number of fronts that exist. The illustration of dominancy based on fronts is depicted in Figure 4.1. Each individual in each front is assigned this front value (rank or fitness). So in the first front individuals are assigned the rank value 1, in the second – 2 and so on.

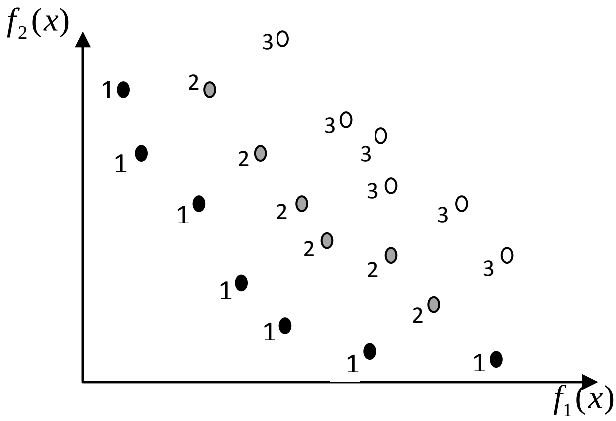

Figure 4.1. Front-based dominancy

In addition, each individual has another parameter called *crowding distance*, which measures the distance between its neighbors. The large crowding distance will give a better diversity. From the selected parents offspring are generated from crossover and mutation operators. While performing all the steps in the algorithm we also check the constraint of eliminating the repeated individuals in the chromosomes, since our problem is an integer programming problem and there should not be same individuals in the same chromosome, because as was mentioned in chapter 2, the customer must be served once and by one vehicle only, so due to this constraint we have to eliminate same individuals in a chromosome. The flowchart of NSGA-II is given in Figure 4.2 [1].

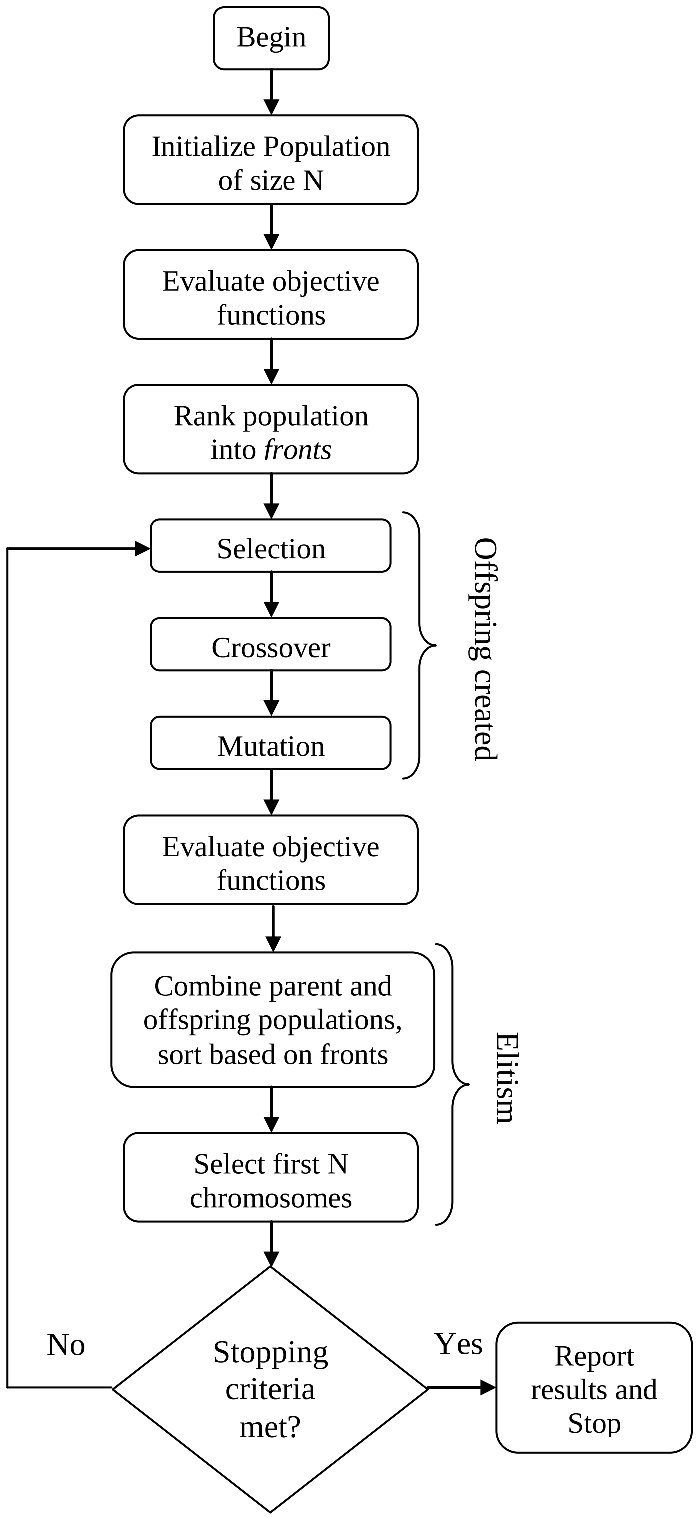

Figure 4.2. Flowchart of NSGA-II

4.1.1 Chromosome Representation and Data Structures

In this algorithm a chromosome is represented as follows. If we consider *n* customers and *k* objectives to be optimized, then in the chromosome the first *n* elements are the individuals. Next, from $n + 1$ to $n + k$ a chromosome contains the objective values. The value at $n + k + 1$ is the front value (rank) and the last value at $n + k + 2$ is the crowding distance. The Figure 4.3 illustrates the representation of a chromosome and the *q* number of vehicles, each serving a particular number of customers depending on their demands and other objective functions with constraints.

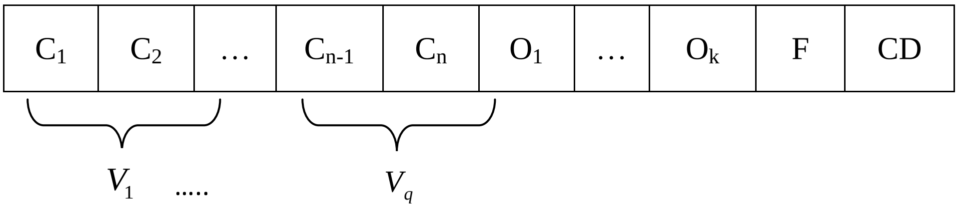

Figure 4.3. Chromosome representation

The data structures used for customers include the following: the coordinates (x,y), demand, earliest start of service time, expected (or desired) start of service time, latest start of service time and the service time.

4.1.2 Genetic Operators

The following genetic operators were used in this algorithm: for selection – binary tournament selection was used, for crossover – simulated binary crossover was used and for mutation – polynomial mutation was used. The tournament selection was described in details in chapter 3, so below the simulated binary crossover and polynomial mutation is explained.

4.1.2.1 Simulated Binary Crossover

The crossover is applied as follows [30, 31]:

$$c_{1,k} = \frac{1}{2}[(1-\beta_k)p_{1,k} + (1+\beta_k)p_{2,k}]$$

$$c_{2,k} = \frac{1}{2}[(1-\beta_k)p_{1,k} + (1-\beta_k)p_{2,k}]$$

Where $c_{i,k}$ is the i^{th} child with k^{th} component, and $p_{i,k}$ is the selected parent and $\beta_k (\geq 0)$ is a sample from a random number generated having the density as follows:

$$p(\beta) = \frac{1}{2}(\eta_c + 1)\beta^{\eta_c}, \text{ if } 0 \leq \beta \leq 1$$

$$p(\beta) = \frac{1}{2}(\eta_c + 1)\frac{1}{\beta^{\eta_c+2}}, \text{ if } \beta > 1$$

Above distribution is obtained from a uniformly sampled random number u between (0,1), and η_c is the distribution index for crossover, which determines how well spread the children will be from selected parents.

$$\beta(u) = (2u)^{\frac{1}{(\eta+1)}}$$

$$\beta(u) = \frac{1}{[2(1-u)]^{\frac{1}{(\eta+1)}}}$$

4.1.2.2 Polynomial Mutation

The polynomial mutation is applied as follows [31, 32]:

$$c_k = p_k + (p_k^u - p_k^l)\delta_k$$

Where c_k is the offspring and p_k is the selected parent with p_k^u and p_k^l being the upper and the lower bounds respectively on the parent component. δ_k is a small variation that is calculated from a polynomial distribution by using the following formula:

$$\delta_k = (2r_k)^{\frac{1}{\eta_m+1}} - 1, \text{ if } r_k < 0.5$$

$$\delta_k = 1 - [2(1 - r_k)]^{\frac{1}{\eta_m + 1}}, \text{ if } r_k \geq 0.5$$

Where r_k is a uniformly sampled random number between (0, 1) and η_m is mutation distribution index.

However, in this thesis, after attempting the simulated binary crossover and polynomial mutation the results were not very good, since our problem is an integer programming problem. Therefore, later we used for crossover and mutation a two-point crossover and insertion mutation respectively.

4.2 Implemented Objective Functions

4.2.1 Load Capacity

The load capacity objective function is calculated with respect to two other parameters, which are load-focusing degree and space utility [26]. So we consider it as one objective function, where load-focusing degree – α and space utility – β are to be maximized in order to get the best loading performance as shown below:

L = 0.5×α + 0.5×β

Where,

$$\alpha = \frac{(N - N')}{N} \qquad \beta = \frac{\sum_{i=1}^{N'} f_i / F_i}{N'}$$

N : the total number of vehicles

N' : the number of vehicles used

α: load-focusing degree – normalization (between 0 – 1)

β: space-utility of vehicle

f_i : the total weight of the commodities for customers served by vehicle i

F_i : loading capacity of vehicle i

4.2.2 Distance Travelled

The minimum distance travelled by vehicle *i* can be determined through minimizing the value of D_i , which is the distance travelled by a vehicle *i*.

$$D_i = \frac{\sum_{j=1}^{n_i+1} (d_{ij} / d_{\max})}{(n_i + 1)}$$

Where,

n_i : the number of customers served by vehicle *i*.

d_{ij} : the distance between the j^{th} customer and the $(j-1)^{th}$ customer by vehicle *i*.

$d_{\max}$: the maximum distance among the all pair vise distances between customers served by all vehicles.

4.2.3 Service Satisfaction

Service satisfaction of a customer is based on concept of fuzzy due time where membership function of fuzzy time corresponds to the grade of satisfaction of a service time [29]. And the objective is to maximize the satisfaction grade. The main approach considers the customers' period of time, in which the grade of satisfaction of service $\mu_i(t)$ is defined as follows:

$$\mu_i(t) = \begin{cases} 1, \text{ if } e_i \le t \le l_i \\ 0, \text{ otherwise.} \end{cases}$$

Where (e_i, l_i) represents the earliest and the latest start time of a customer *i*, and *t* is the time a customer *i* is served with t>0. The illustration of fuzzy due time approach for maximizing the satisfaction grade of service time is depicted in Figure 4.5.

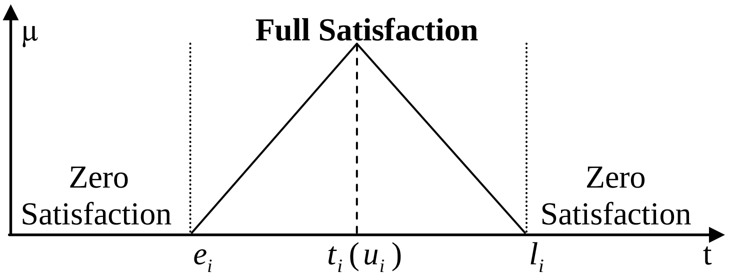

Figure 4.4. Fuzzy due time

Where (e_i, u_i, l_i) represents the earliest, expected and the latest start time of customer *i*, and t_i is the time at which a customer *i* is served. The following function implements the fuzzy due time for satisfaction grade of service time:

$$\mu_i(t_i) = \begin{cases} 0, & \text{if } t_i < e_i \\ \dfrac{(t_i - e_i)}{(u_i - e_i)}, & \text{if } e_i \le t_i \le u_i \\ \dfrac{(l_i - t_i)}{(l_i - u_i)}, & \text{if } u_i \le t_i \le l_i \\ 0, & \text{if } t_i > l_i \end{cases}$$

4.2.4 Waiting Time

In this thesis, two different approaches were implemented to minimize the waiting time, and this is the first approach.

Minimization of waiting time helps us to increase the satisfaction grade of service time [27, 29]. The relationship of service time between two customers *i* and j of a vehicle is illustrated in Figure 4.6. The vehicle finishes serving customer *i* at t_i and then it takes time q_{ij} for vehicle to travel to customer *j* and starts serving this customer at t_j. The waiting time w_k occurs when the vehicle k arrives, for serving customer *j* at t_j, earlier than the earliest time of customer *j* e_j.

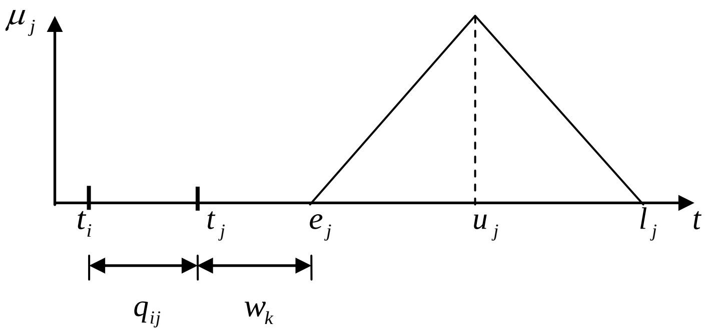

Figure 4.5. Fuzzy due time and waiting time

The waiting time w_k can be calculated as follows:

$w_k = e_j - t_j$, if $t_j < e_j$.

Where

$t_j = t_i + q_{ij} + s_i$

t_j : the start of service time of customer *j*

t_i : the start of service time of customer *i*

q_{ij} : the time to travel from customer *i* to customer *j*

s_i : the service time of customer *i*

Now, we can calculate the total waiting time of a vehicle *j* serving customer *k*. The minimum average waiting time W_j can be obtained as follows:

$$W_j = 1/n' \sum_{k=1}^{n'_j} (w_{jk} / w_{\max}) .$$

Where

n’: the number of customers for whom the vehicle *j* was waiting

w_{jk} : the waiting time of vehicle *j* serving customer *k*

$w_{\max}$: the maximum waiting time of a vehicle among the customers.

4.2.5 Delay Times

In this section, the main approach is considered. Delay times, or in other words, the waiting time of vehicles and the waiting time of customers are discussed here with a different approach than the one described above. As was mentioned before, fuzzy due time was applied on the time window of customers, which enabled us to maximize the satisfaction grade of customers. So, here we apply the same principle to the windows of vehicles, which we created before and the after the time window of a customer, and we call them *delay times* of vehicles, since these time windows belong to vehicles. This approach minimizes the waiting time of customers and thus increases the satisfaction grade of customers as well. In Figure 4.7 the illustration of fuzzy due time for delay times is depicted.

We create some period of time (can be random) before the earliest and after the latest service times of a customer. If the vehicle arrives before the earliest time to arrive **X**, then the first delay time of this vehicle is maximum, i.e. 1, or if it arrives between **X** and e_j , then first delay time is between 0 and 1, otherwise it is 0. Also in second delay time (i.e. waiting time of a customer), if arrival of a vehicle is after **Y**, then second delay time is 1, or if arrival is between l_j and **Y**, then it is between 0 and 1, otherwise it is 0.

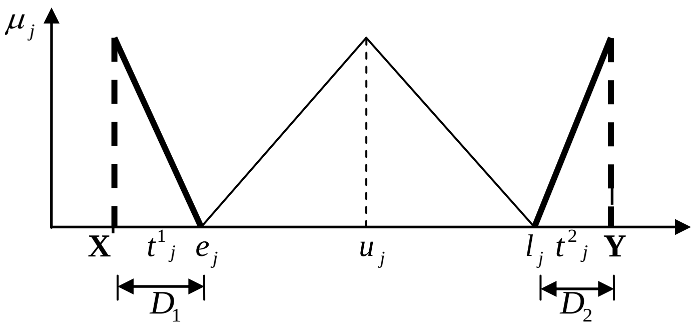

Figure 4.6. Fuzzy due time with delay times

Where,

$\mu_j^1 = (e_j - t_j^1)/(e_j - \mathbf{X})$ and $\mu_j^2 = (t_j^2 - l_j)/(\mathbf{Y} - l_j)$

X: the earliest time for vehicle *k* to arrive for serving customer *j*, X$<e_j$.

Y: the latest time for vehicle *k* to arrive for serving customer *j*, Y$>l_j$.

D_1 : the delay time of vehicle *k* before the earliest service time of customer *j* (i.e. the waiting time of vehicle *k*).

D_2 : the delay time of vehicle *k* after the latest service time of customer *j* (i.e. the waiting time of a customer *j*).

Chapter 5

EXPERIMENTAL RESULTS

5.1 Fuzzy VRP Instances

In this thesis, for experimental results and solutions six VRPTW Benchmark Problems were used, also known as Solomon Benchmark Problems which are: R1-type, C1-type, RC1-type, R2-type, C2-type and RC2-type. These six problems are geographical data, the number of customers serviced by a vehicle, percent of time-constrained customers and tightness and positioning of the time windows. In problems R1-type and R2-type the geographical data are randomly generated. In problems C1-type and C2-type the geographical data are clustered. RC1-type and RC2-type problem sets include previous types of problems. Therefore, they are a mix of random and clustered structures. These problem sets are divided into two categories: short scheduling horizon and a long scheduling horizon. Short scheduling horizon problems allow only a few customers per route around 5-10 customers, whereas a long scheduling horizon problems allow many customers around 30 or more. Problem sets R1, C1 and RC1 have a short scheduling horizon, but R2, C2, RC2 have a long scheduling horizon.

The widths of time windows in problems are different, whereas the coordinates of customers are same for all problems within one type – R, C and RC. Some of the problems have very tight time windows. For example, customers may be less tolerable for earliness and even also for lateness or only for earliness of for lateness. Moreover, other problems may have time windows hardly constraining.

All of these six problems have the following data structures: the first line is the name of the file, for example, R112, and then the next line gives information on the number of vehicles and their capacity. After that, the data structure is given for customers as follows: customer number, *x* coordinate, *y* coordinate, demand, ready time, due time and the service time.

For experimental results several combinations of objective functions were carried out in order to see the performance of proposed work clearly. Therefore, next sections present results for satisfaction grade of customers and first delay, satisfaction grade of customers and second delay, and then satisfaction grade of customers, first delay and second delay. Moreover, several results for other objective functions will be given in a combination of three of them.

Results for all objective functions have the same unit, which is seconds.

5.2 Satisfaction Grade and First Delay

For satisfaction grade and first delay the experimental results with different population sizes were carried out – 100, 500 and 1000. Although, in general the population size affects the results, the results obtained in this experimental work do not get better as we increase the population size. This is due to the reason of not giving the reasonable number of generations to get better results. The number of generations was small, because it takes huge amount of time to get results. So, the reasonable amount of time was also taken into consideration while obtaining the results.

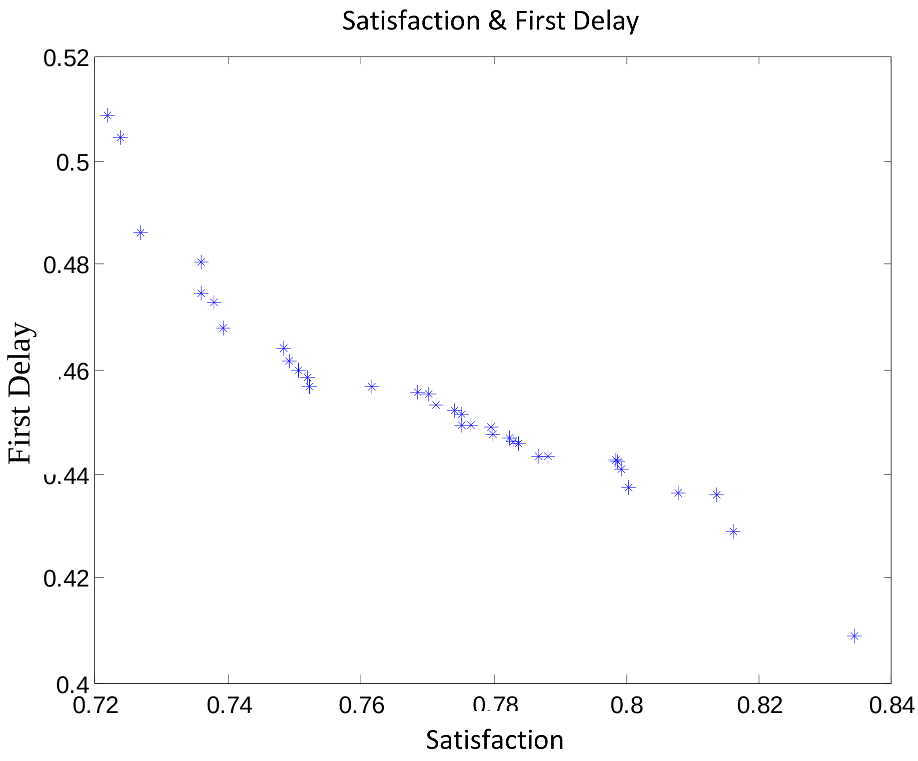

Figure 5.1. Satisfaction grade and first delay, population size 100, file C109

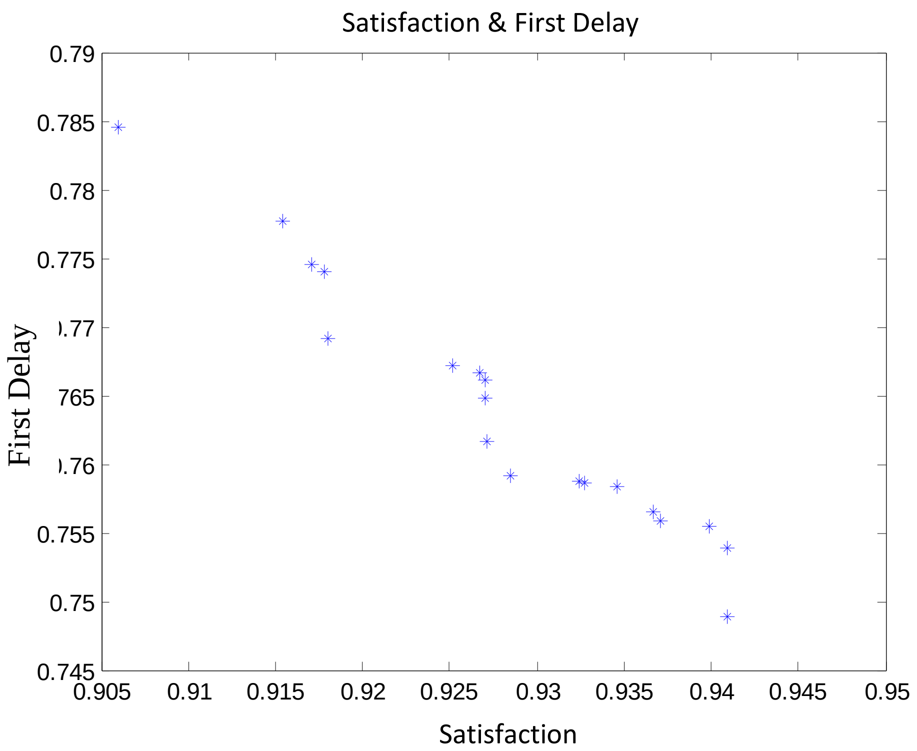

Figure 5.2. Satisfaction grade and first delay, population size 100, file C208

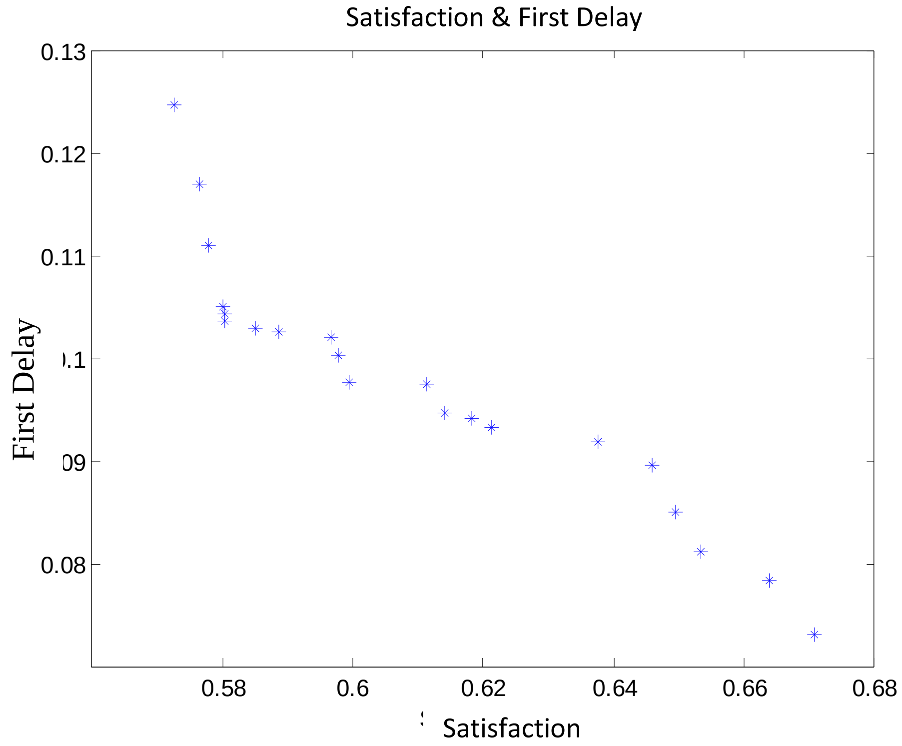

Figure 5.3. Satisfaction grade and first delay, population size 100, file R112

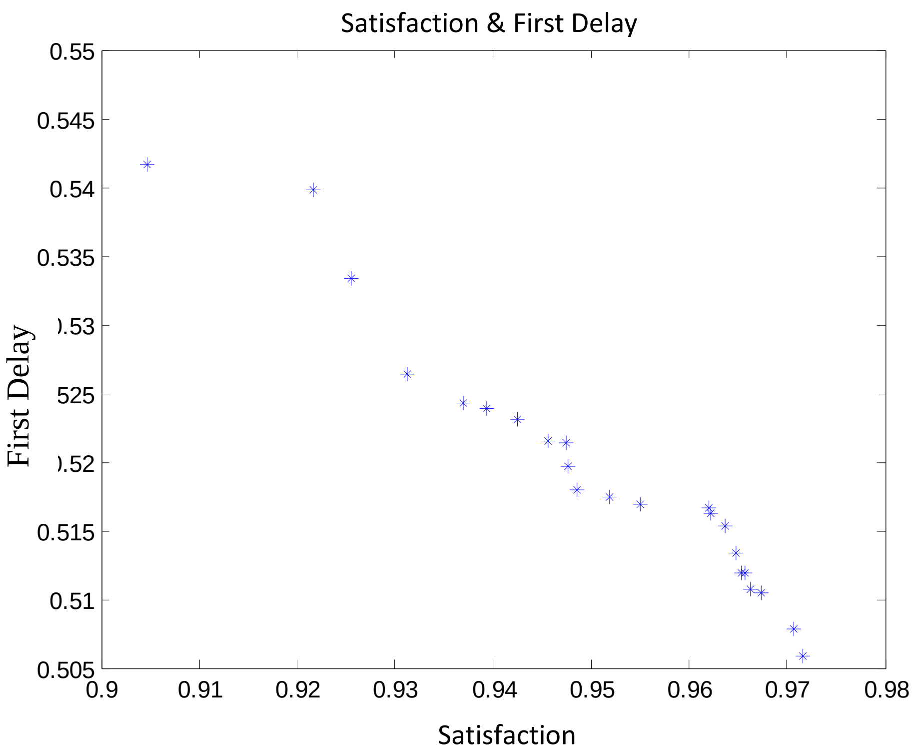

Figure 5.4. Satisfaction grade and first delay, population size 100, file R211

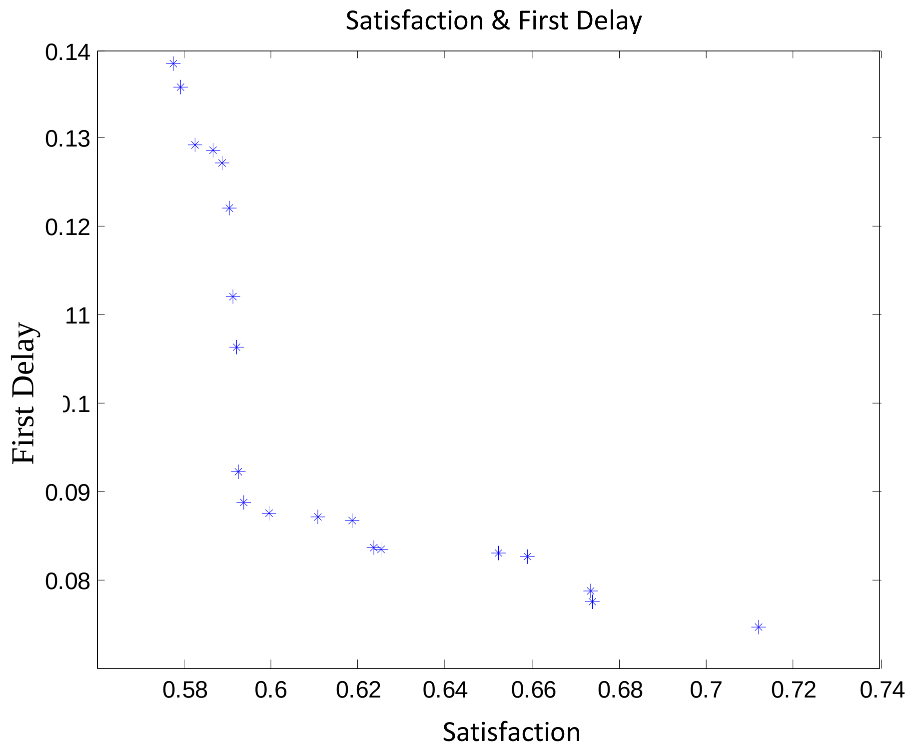

Figure 5.5. Satisfaction grade and first delay, population size 100, file RC108

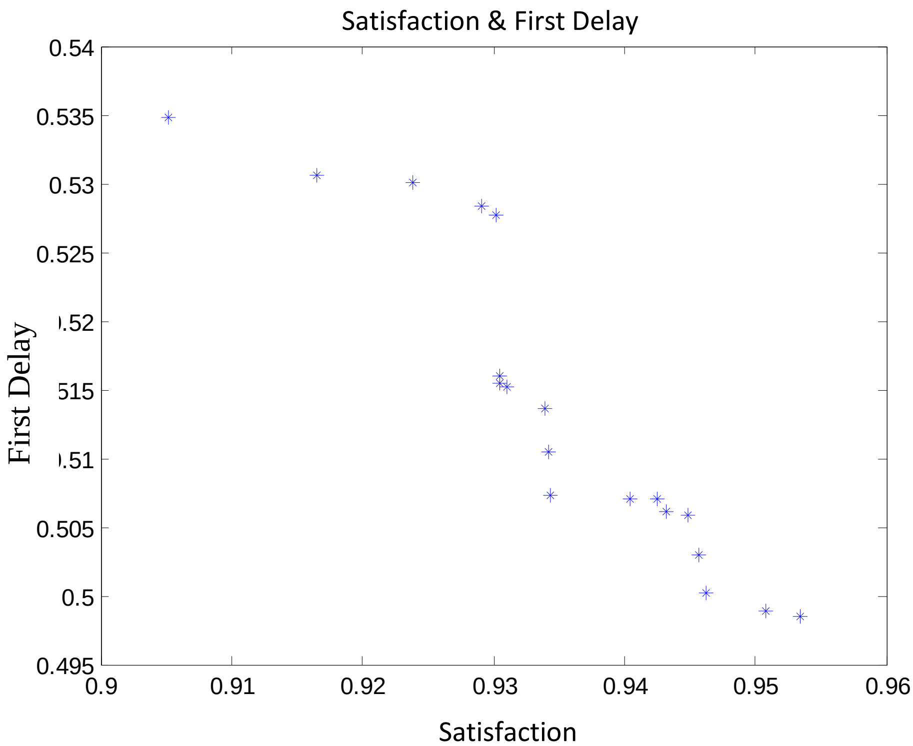

Figure 5.6. Satisfaction grade and first delay, population size 100, file RC208

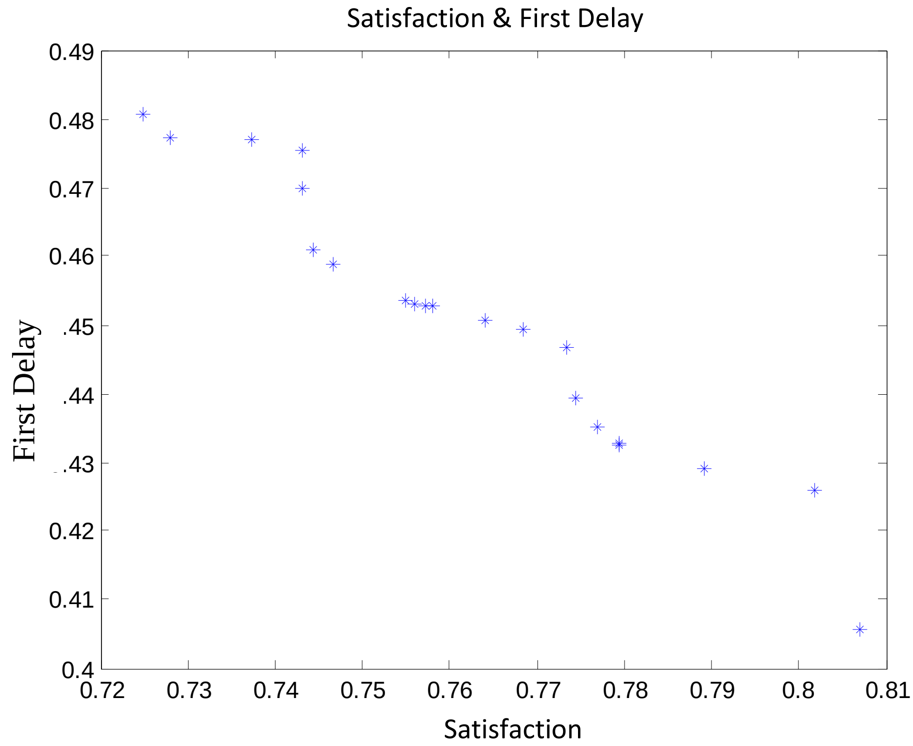

Figure 5.7. Satisfaction grade and first delay, population size 500, file C109

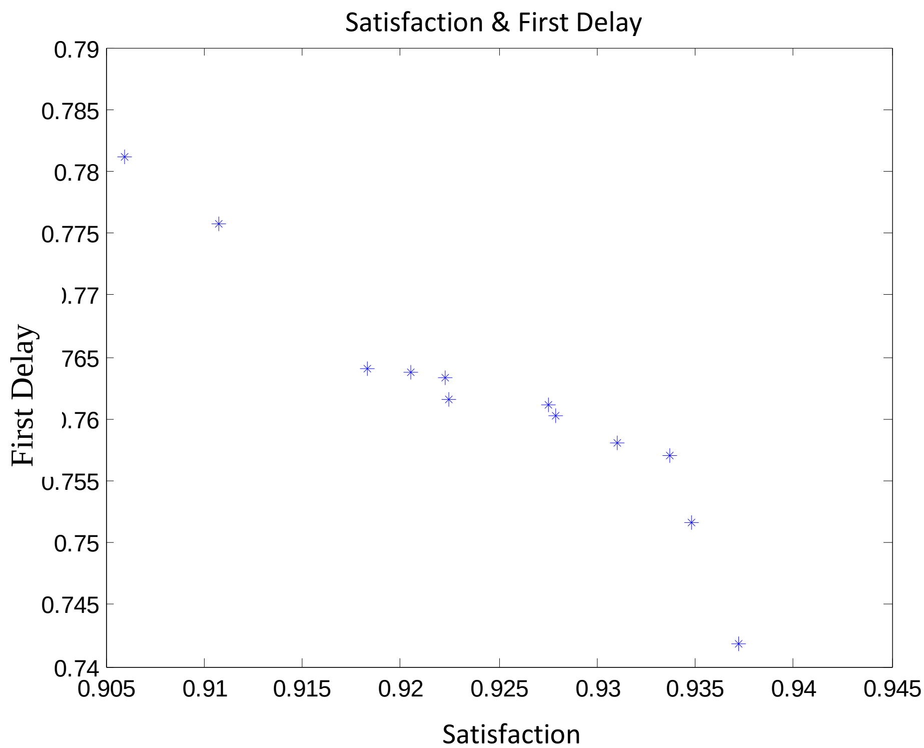

Figure 5.8. Satisfaction grade and first delay, population size 500, file C208

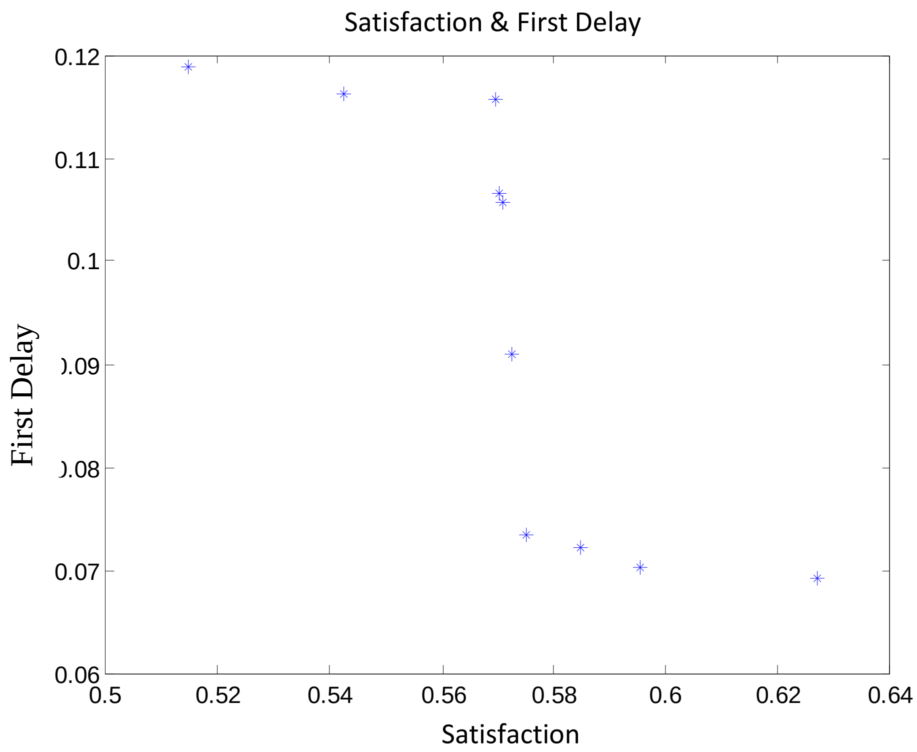

Figure 5.9. Satisfaction grade and first delay, population size 500, file R112

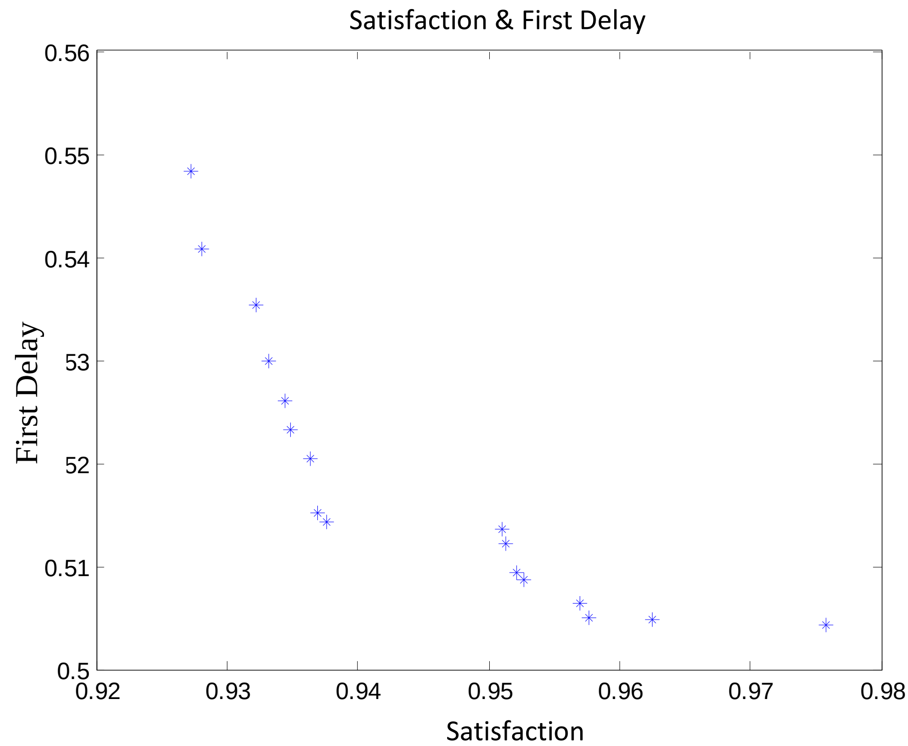

Figure 5.10. Satisfaction grade and first delay, population size 500, file R211

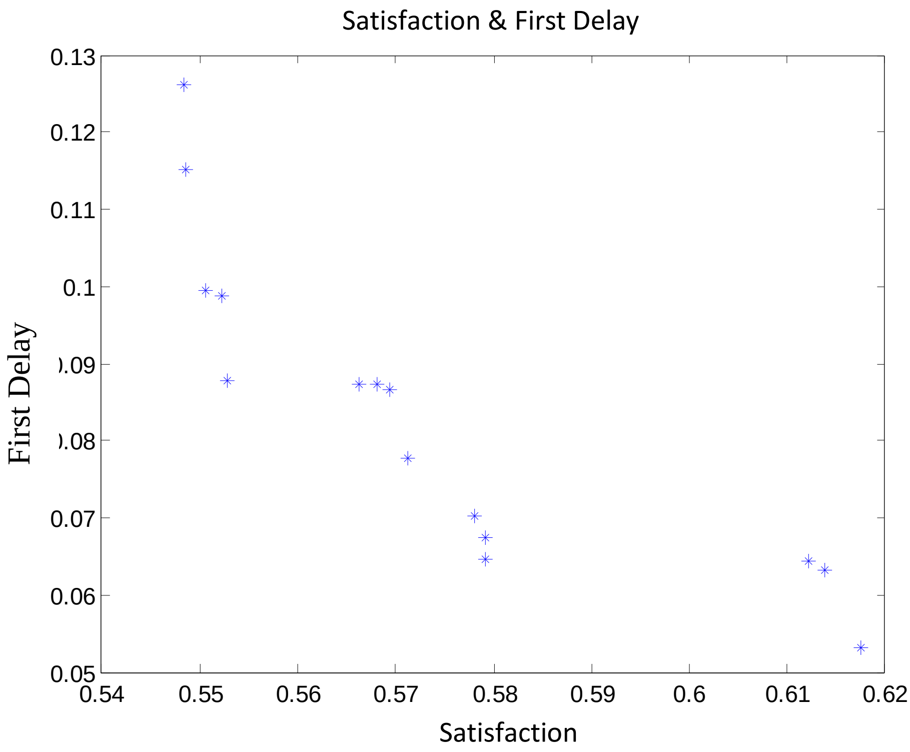

Figure 5.11. Satisfaction grade and first delay, population size 500, file RC108

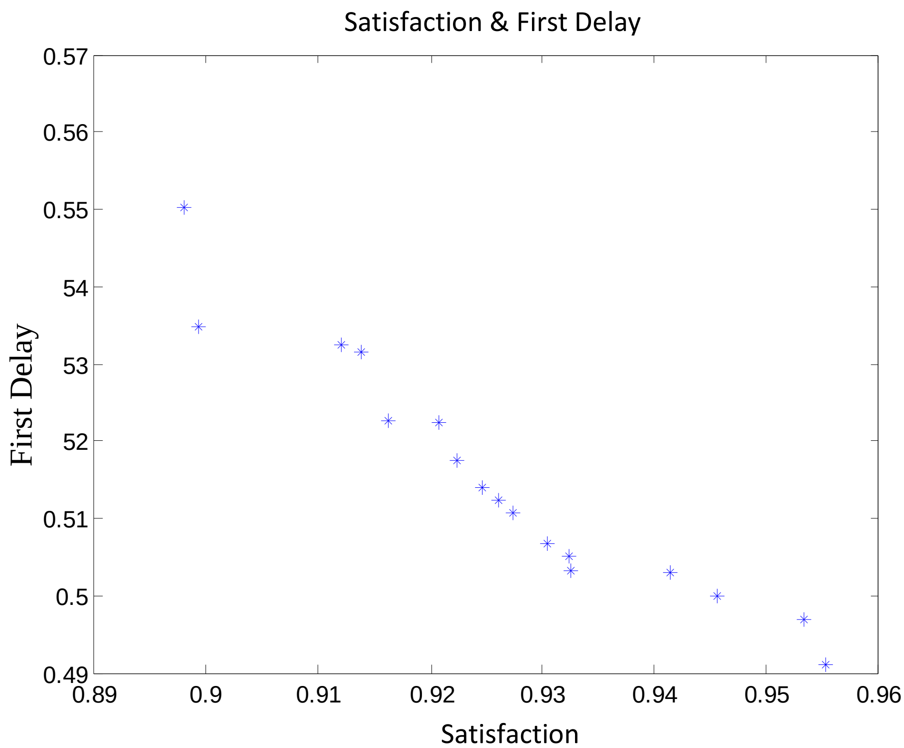

Figure 5.12. Satisfaction grade and first delay, population size 500, file RC208

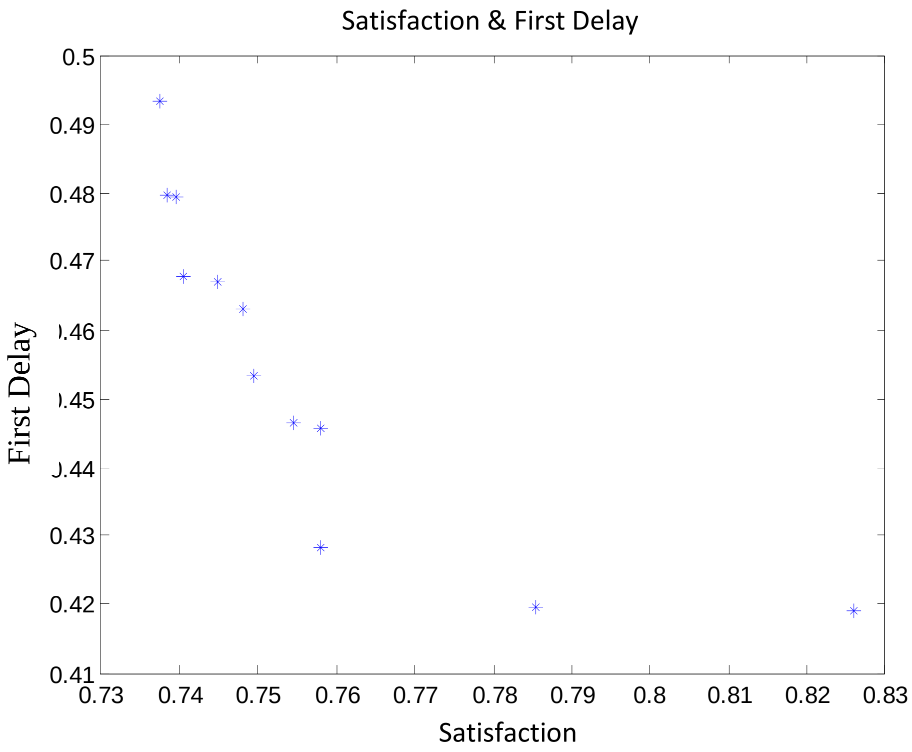

Figure 5.13. Satisfaction grade and first delay, population size 1000, file C109

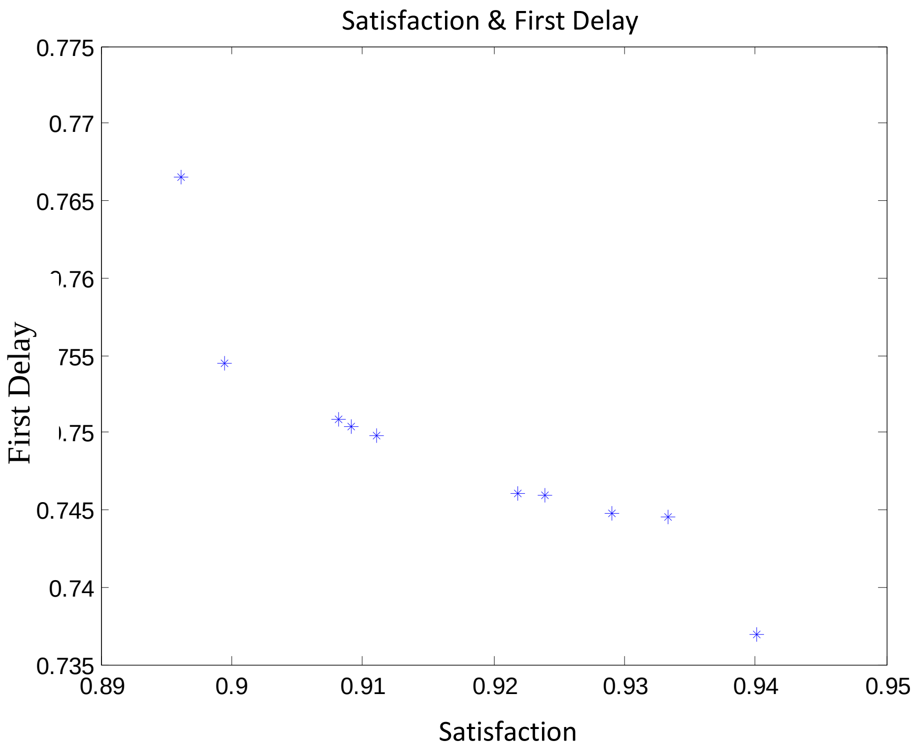

Figure 5.14. Satisfaction grade and first delay, population size 1000, file C208

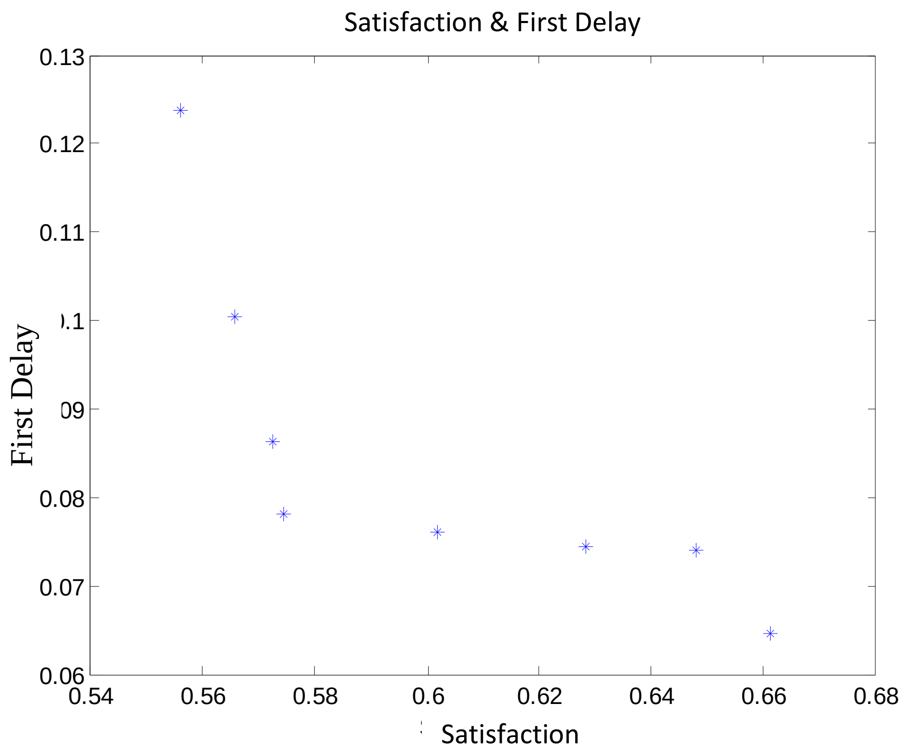

Figure 5.15. Satisfaction grade and first delay, population size 1000, file R112

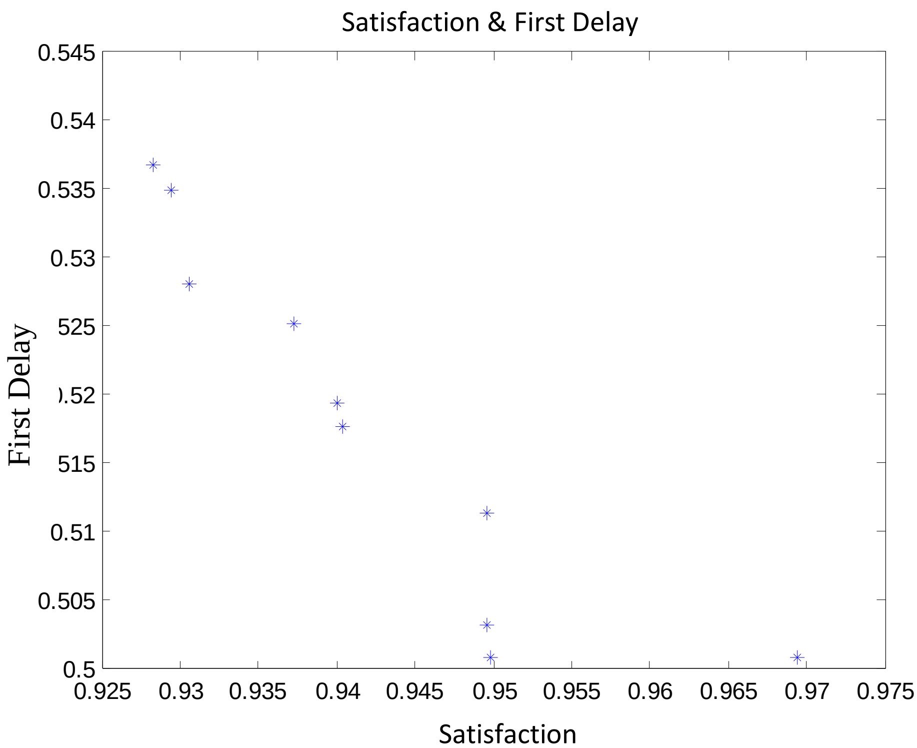

Figure 5.16. Satisfaction grade and first delay, population size 1000, file R211

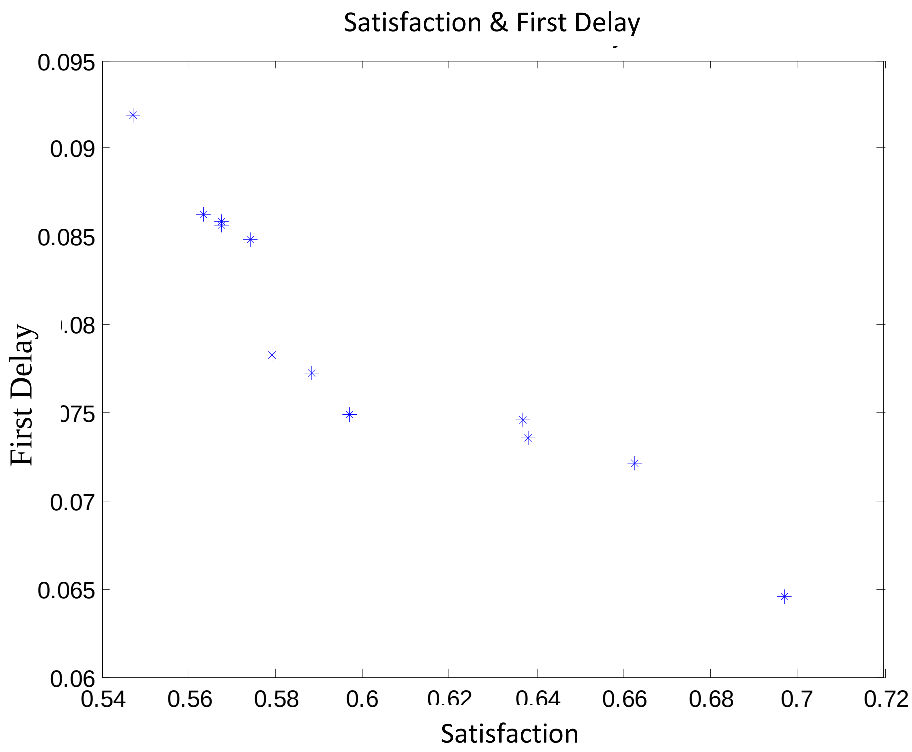

Figure 5.17. Satisfaction grade and first delay, population size 1000, file RC108

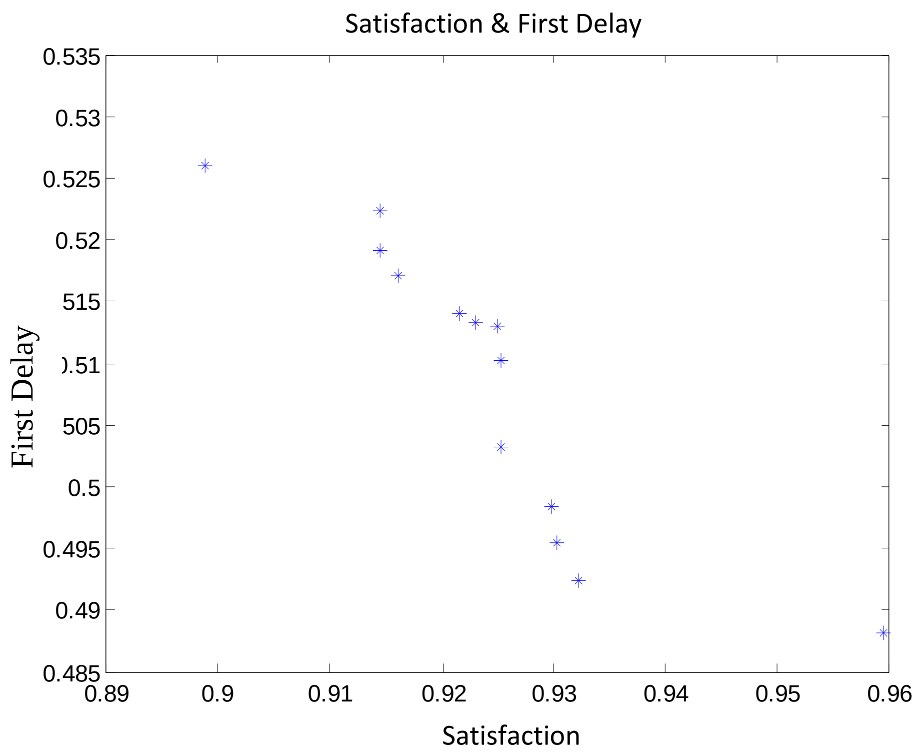

Figure 5.18. Satisfaction grade and first delay, population size 1000, file RC208

Results show that as first delay decreases the satisfaction grade of a customer increases. And this is the reason why first delay must be minimized, since it gives rise to the satisfaction grade objective function.

5.3 Satisfaction Grade and Second Delay

For satisfaction grade and second delay the results were as follows: for population size 100, in C208, R211 and RC208 types of problems the second delay was zero. For population size 500 and 1000, in C109, C208, R211 and RC208 types of problems the second delay was zero. That means for these types of problems the waiting time of customers is zero. The following results are for the rest of the problem types.

In this set of results as second delay decreases the satisfaction grade increases, which is similar to the results above.

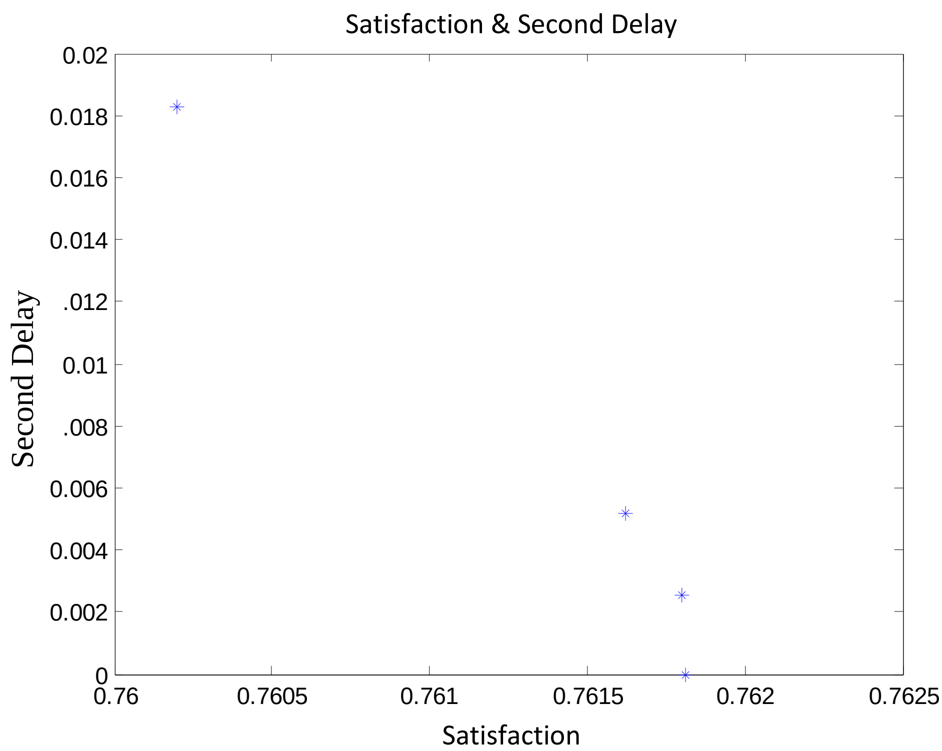

Figure 5.19. Satisfaction grade and second delay, population size 100, file C109

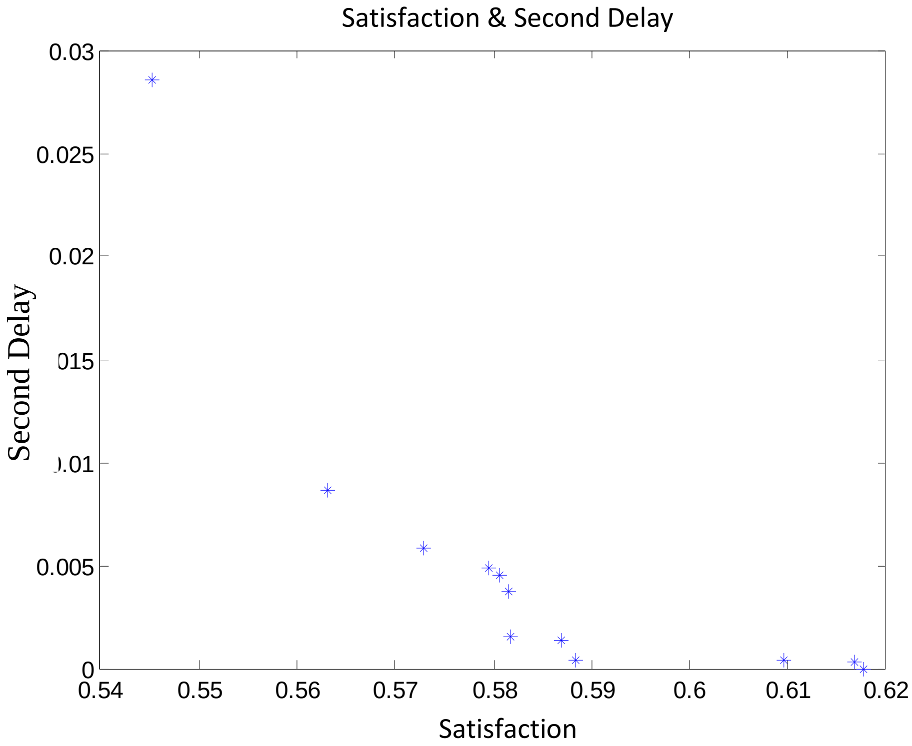

Figure 5.20. Satisfaction grade and second delay, population size 100, file R112

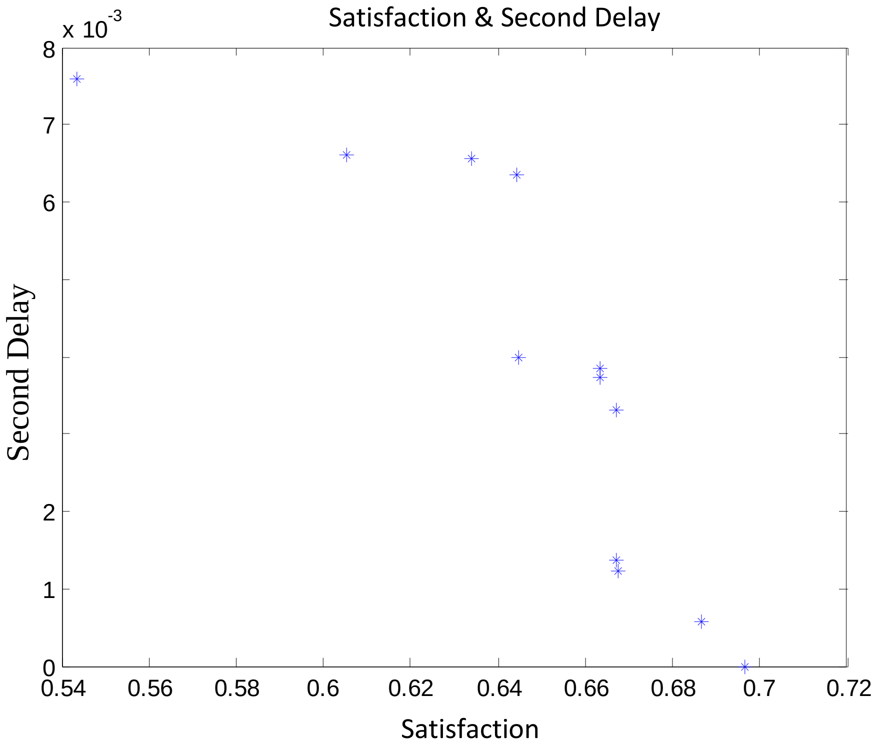

Figure 5.21. Satisfaction grade and second delay, population size 100, file RC108

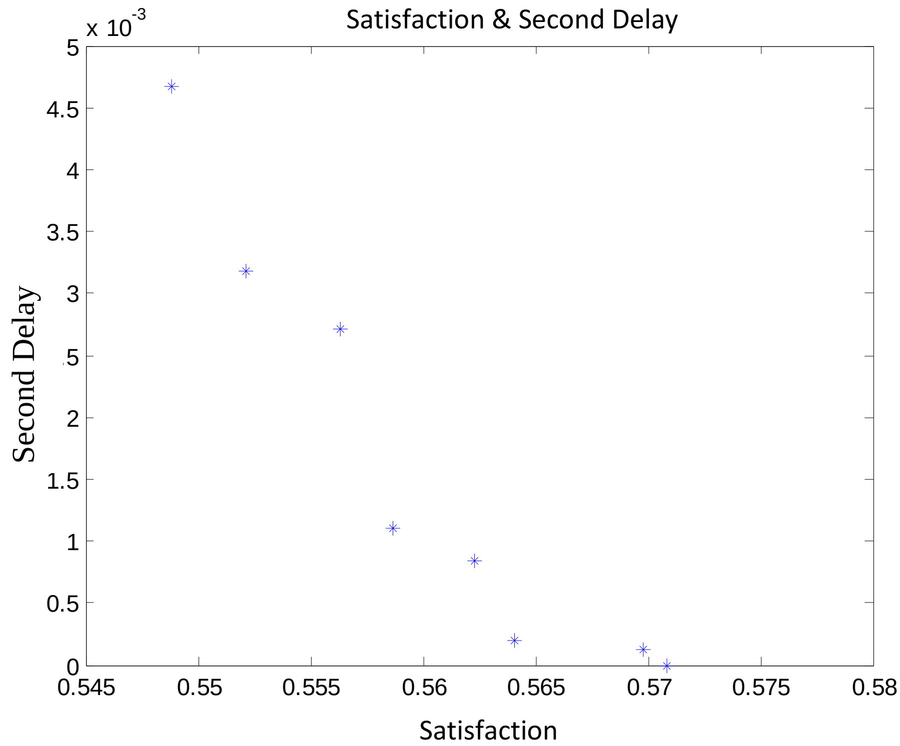

Figure 5.22. Satisfaction grade and second delay, population size 500, file R112

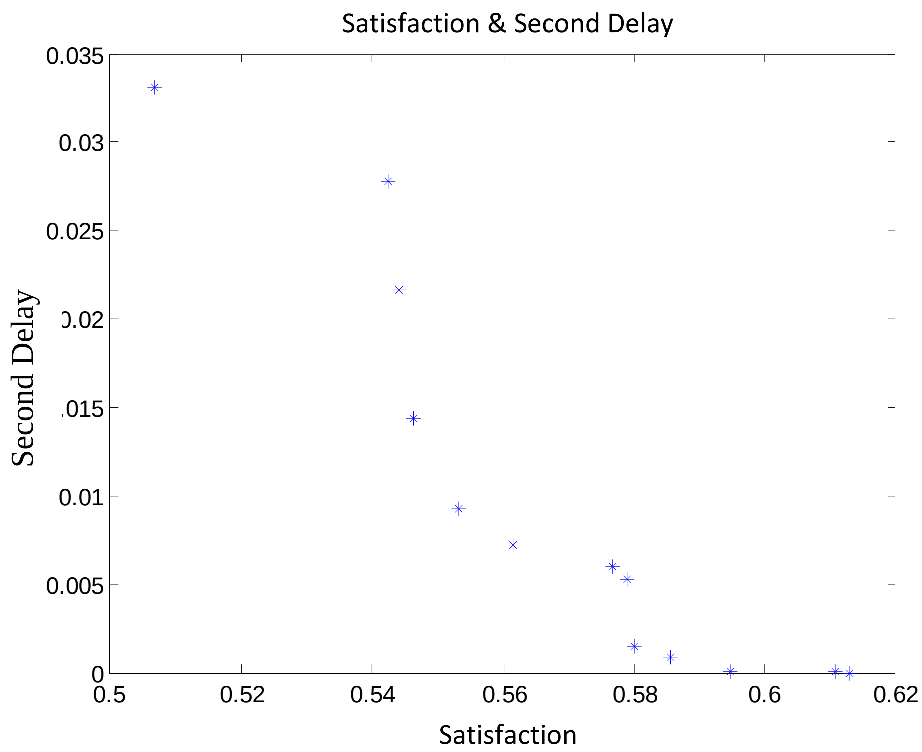

Figure 5.23. Satisfaction grade and second delay, population size 500, file RC108

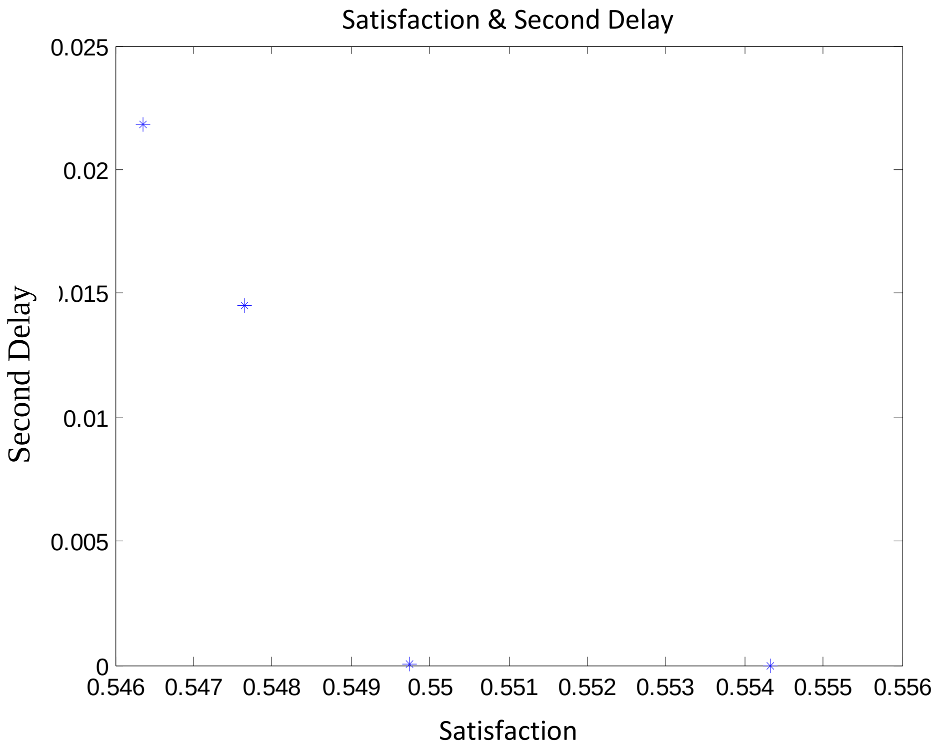

Figure 5.24. Satisfaction grade and second delay, population size 1000, file R112

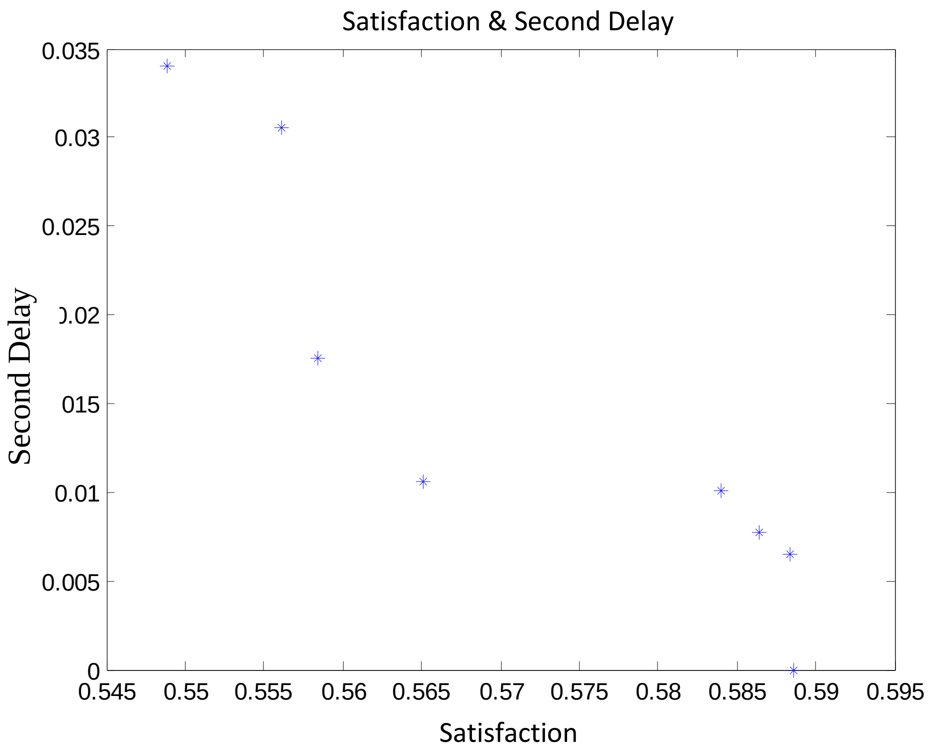

Figure 5.25. Satisfaction grade and second delay, population size 1000, file RC108

5.4 Satisfaction Grade and Both Delays

The following are the results obtained for all three objectives functions. Therefore results are represented in 3-dimnesional space. In this set of experimental results the second delay is zero for all populations in the following types of problems: C208, R211 and RC208.

In the results given below, as second delay and first delay decrease the satisfaction grade increases. So, both second and first delays are minimized in order to maximize the satisfaction grade of customers.

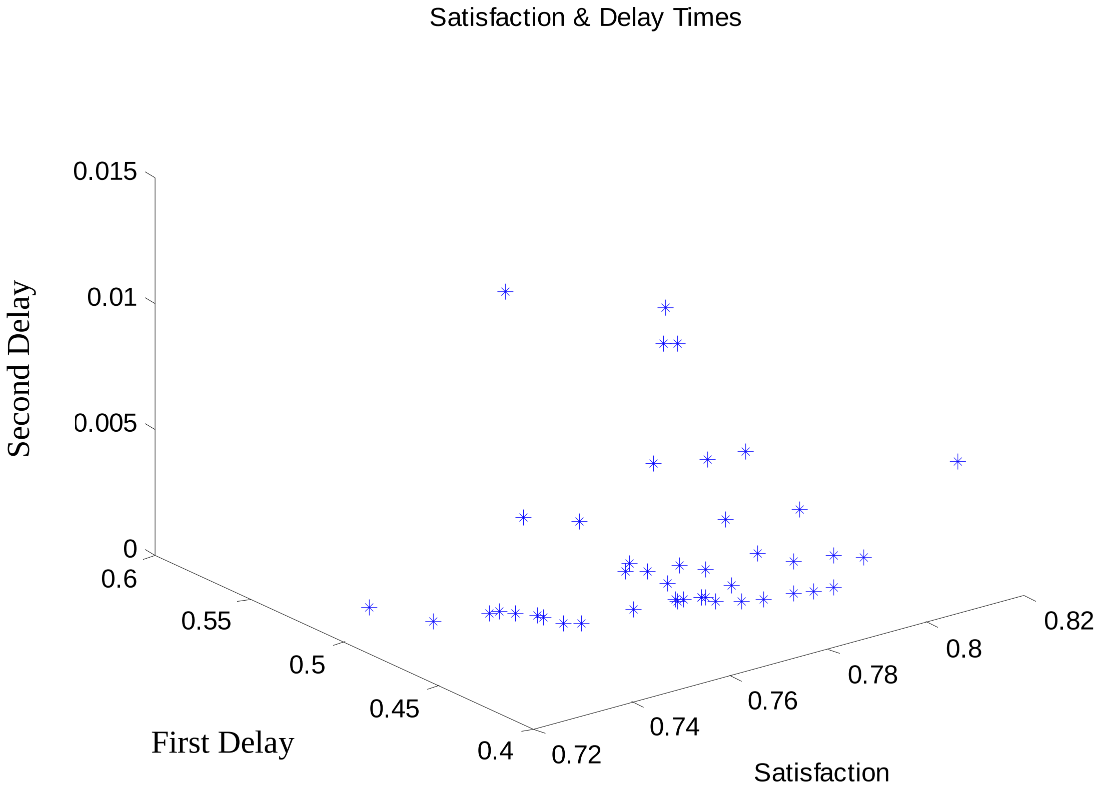

Figure 5.26. Satisfaction grade and both delays, population size 100, file C109

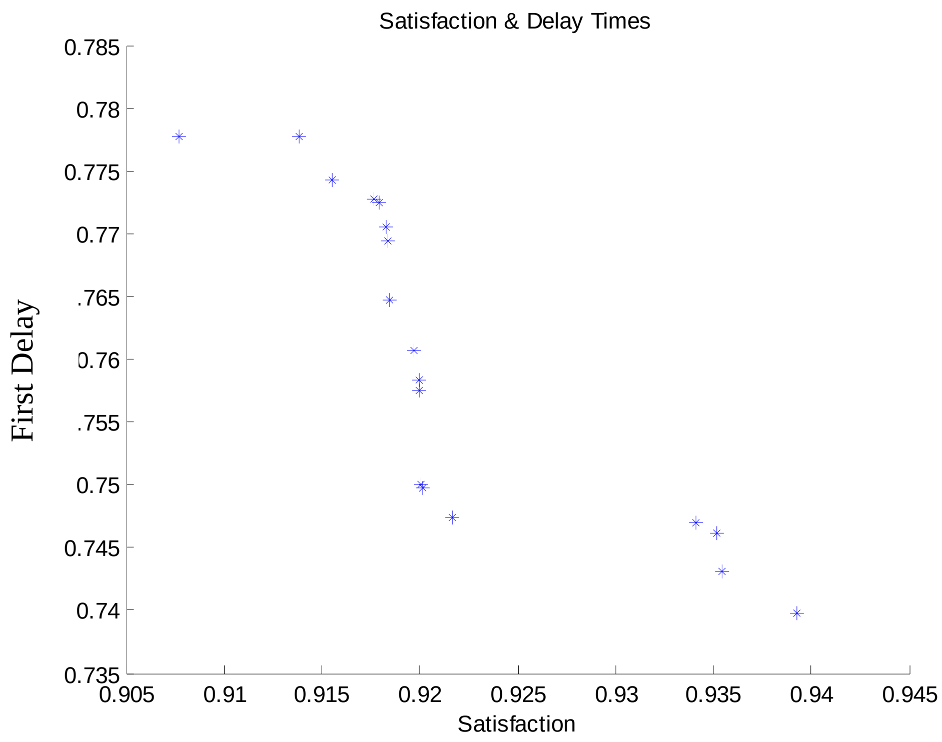

Figure 5.27. Satisfaction grade and both delays, population size 100, file C208

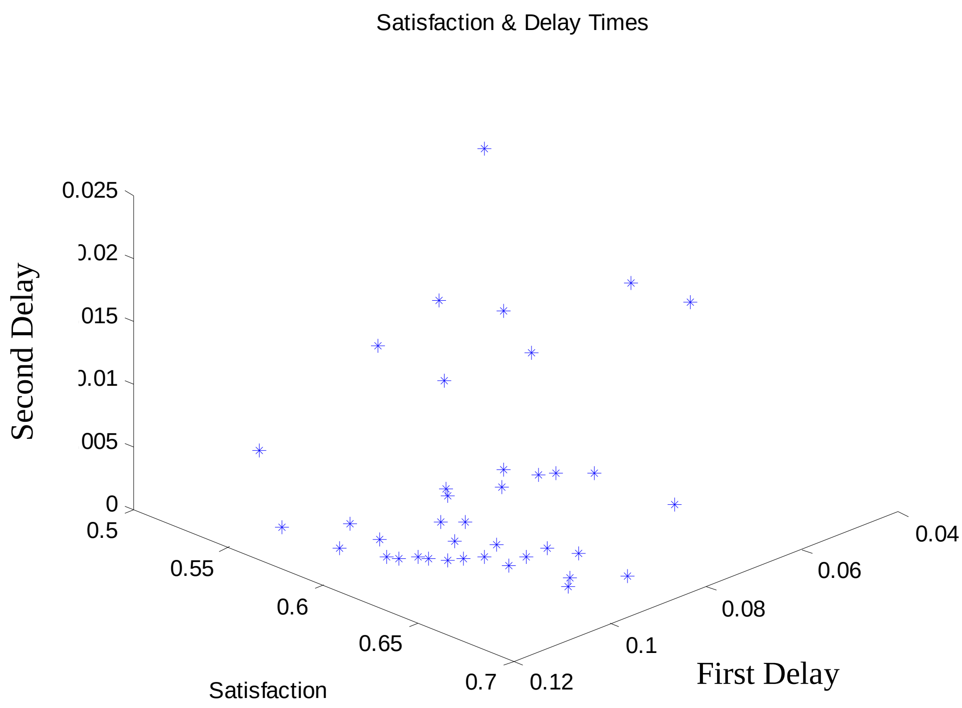

Figure 5.28. Satisfaction grade and both delays, population size 100, file R112

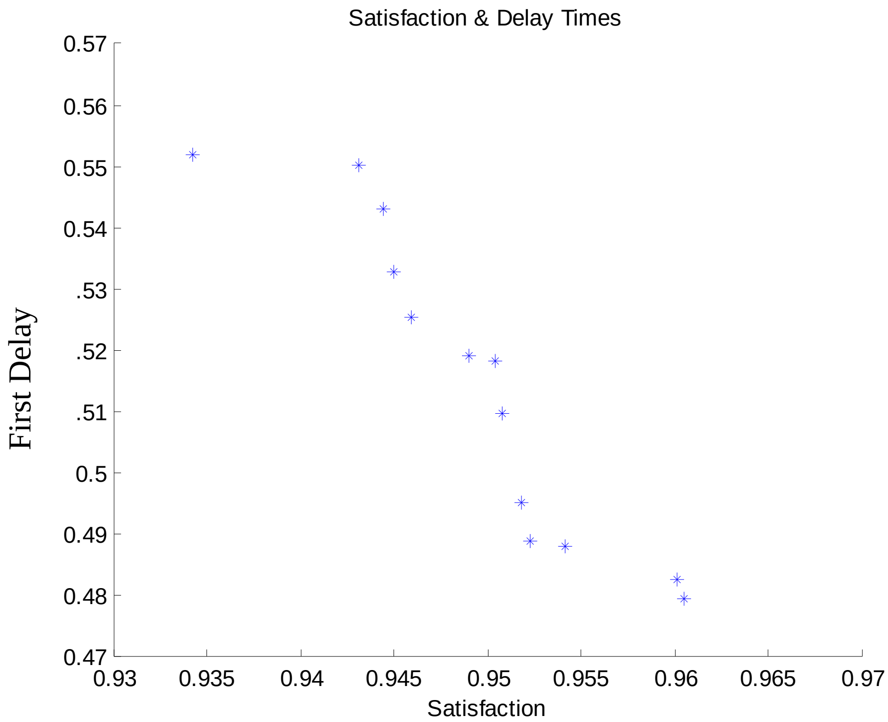

Figure 5.29. Satisfaction grade and both delays, population size 100, file R211

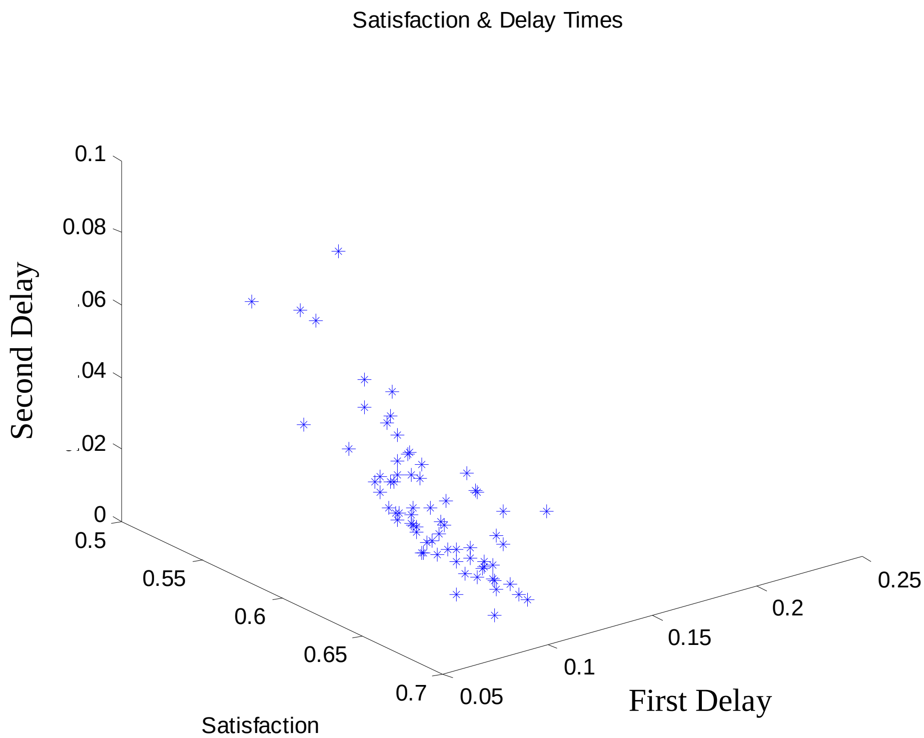

Figure 5.30. Satisfaction grade and both delays, population size 100, file RC108

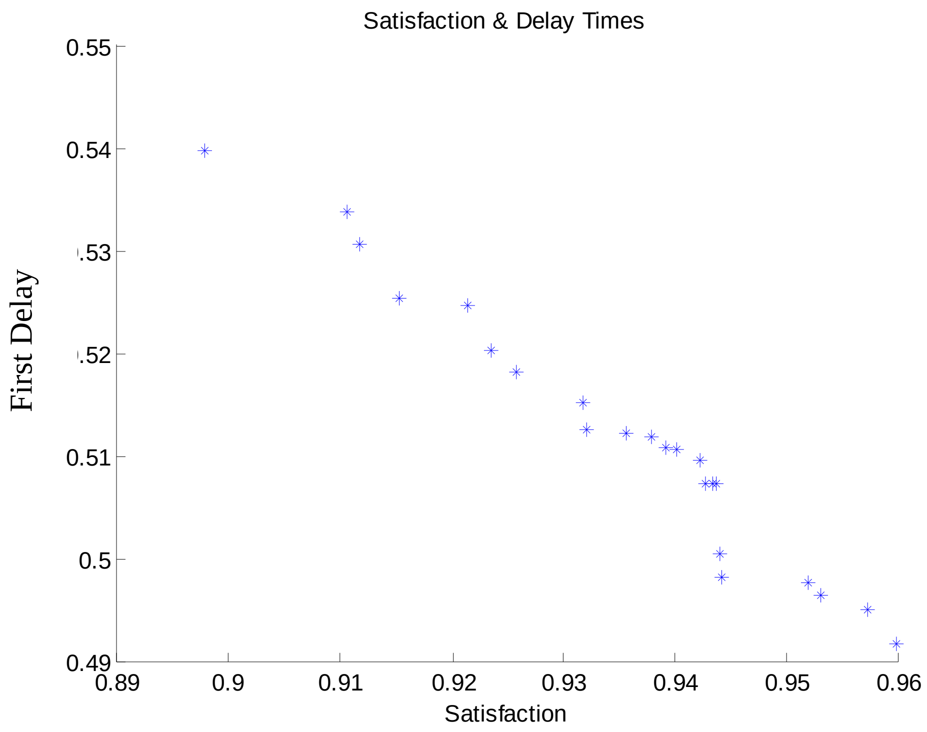

Figure 5.31. Satisfaction grade and both delays, population size 100, file RC208

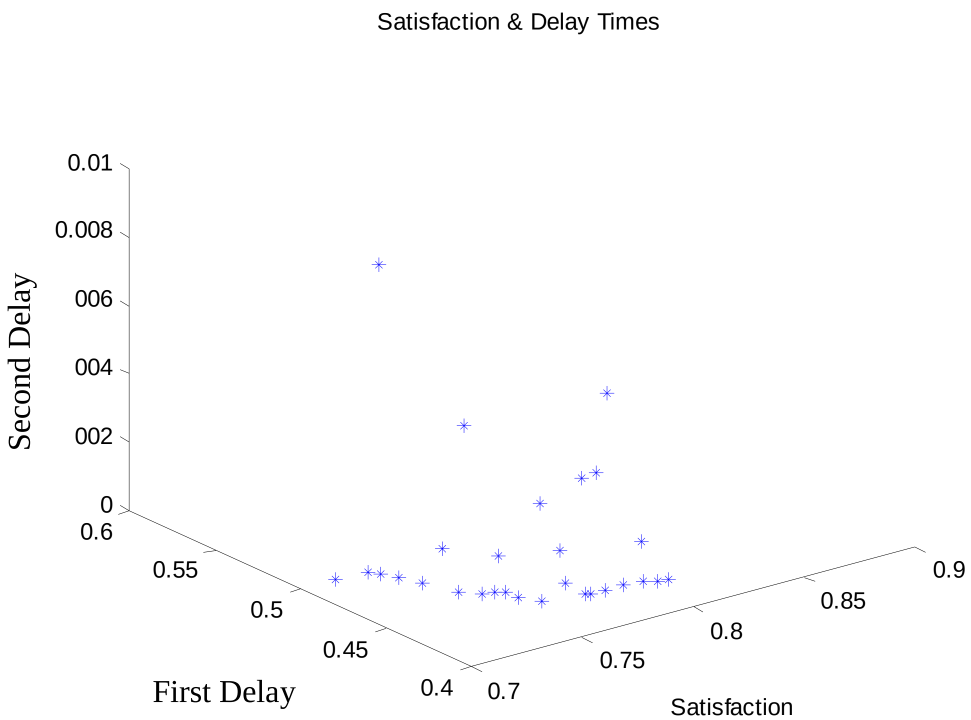

Figure 5.32. Satisfaction grade and both delays, population size 500, file C109

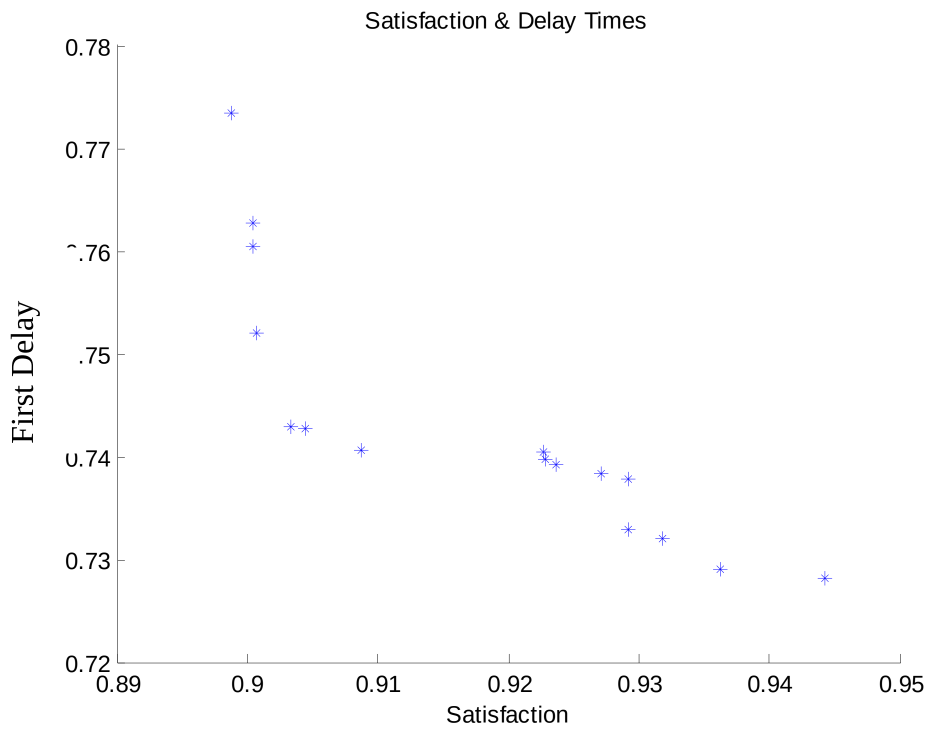

Figure 5.33. Satisfaction grade and both delays, population size 500, file C208

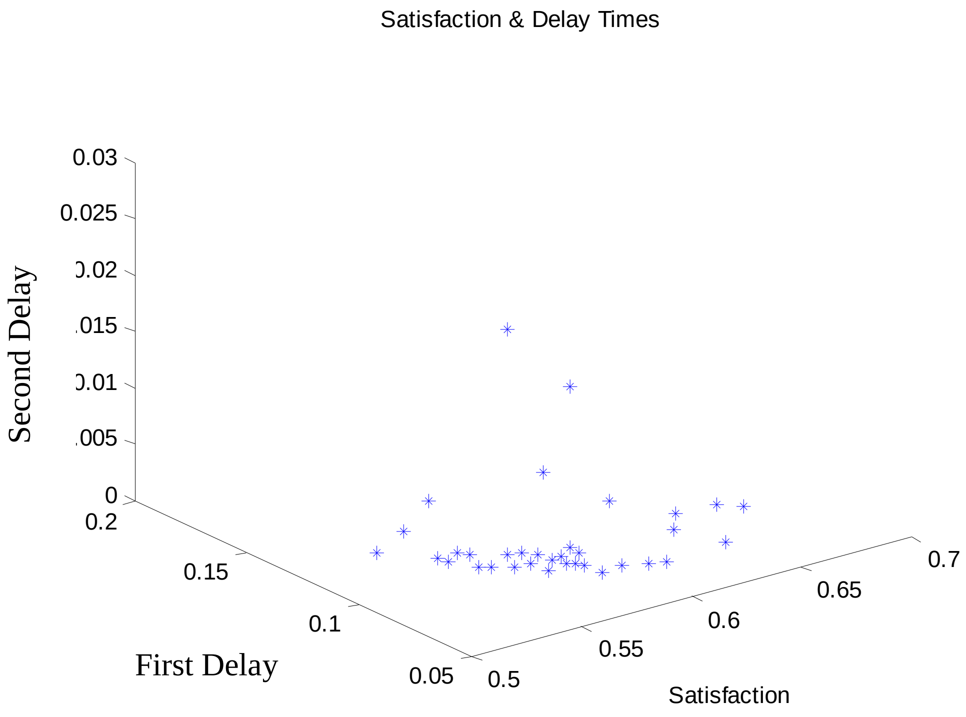

Figure 5.34. Satisfaction grade and both delays, population size 500, file R112

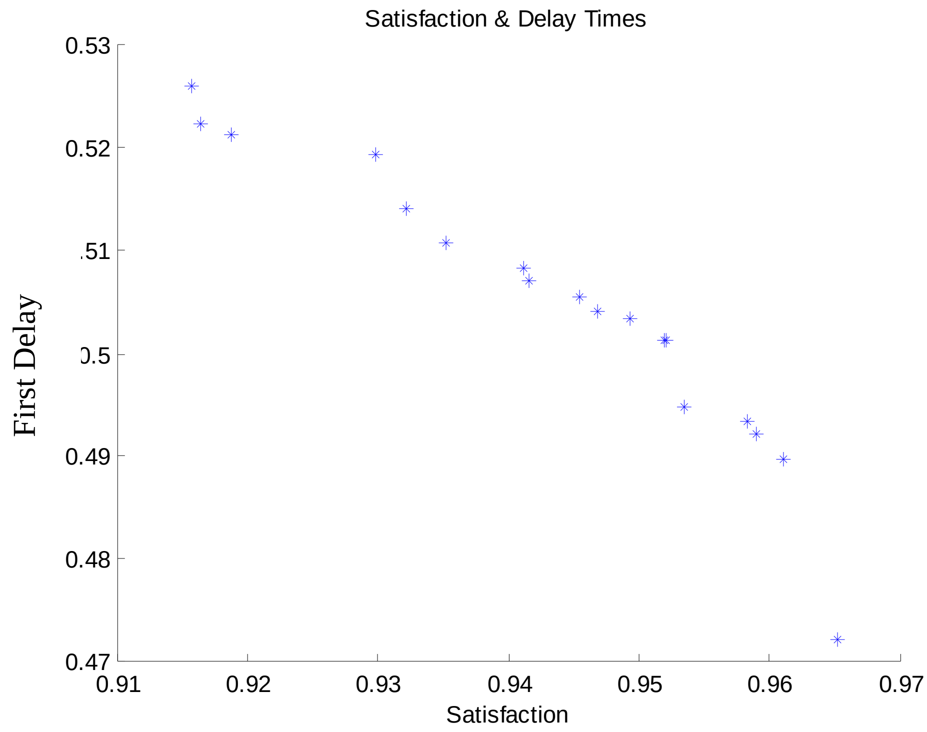

Figure 5.35. Satisfaction grade and both delays, population size 500, file R211

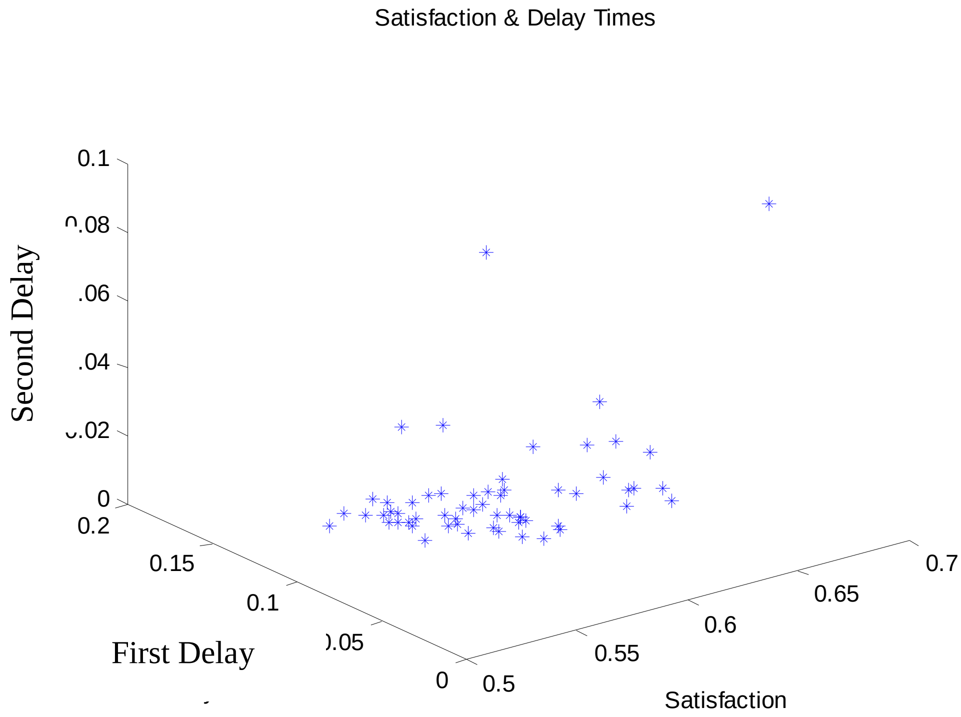

Figure 5.36. Satisfaction grade and both delays, population size 500, file RC108

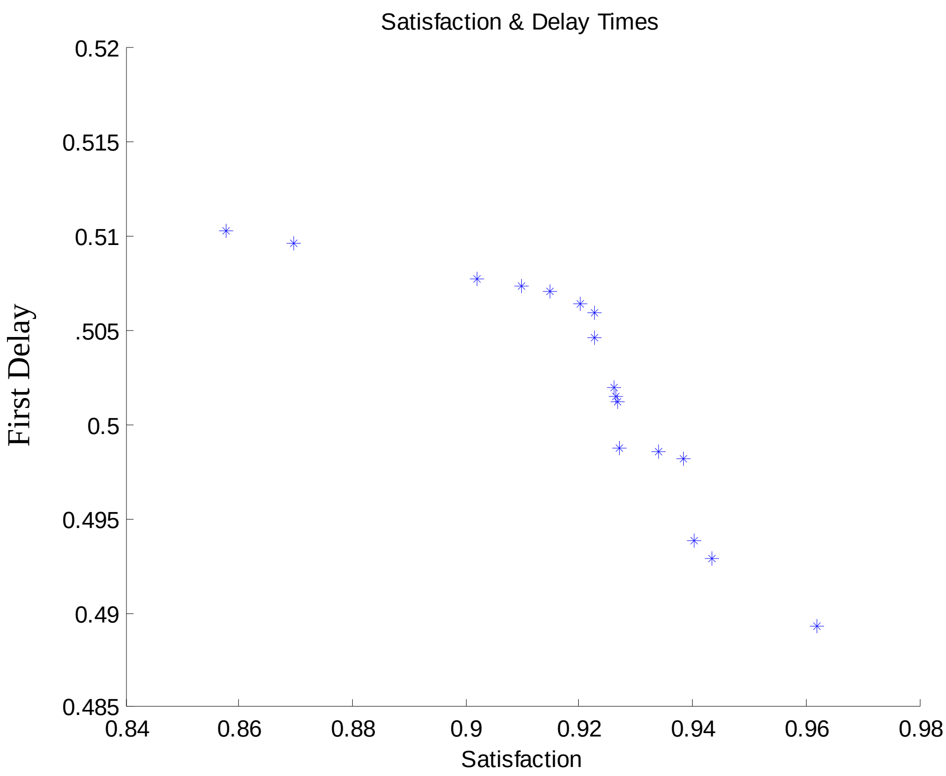

Figure 5.37. Satisfaction grade and both delays, population size 500, file RC208

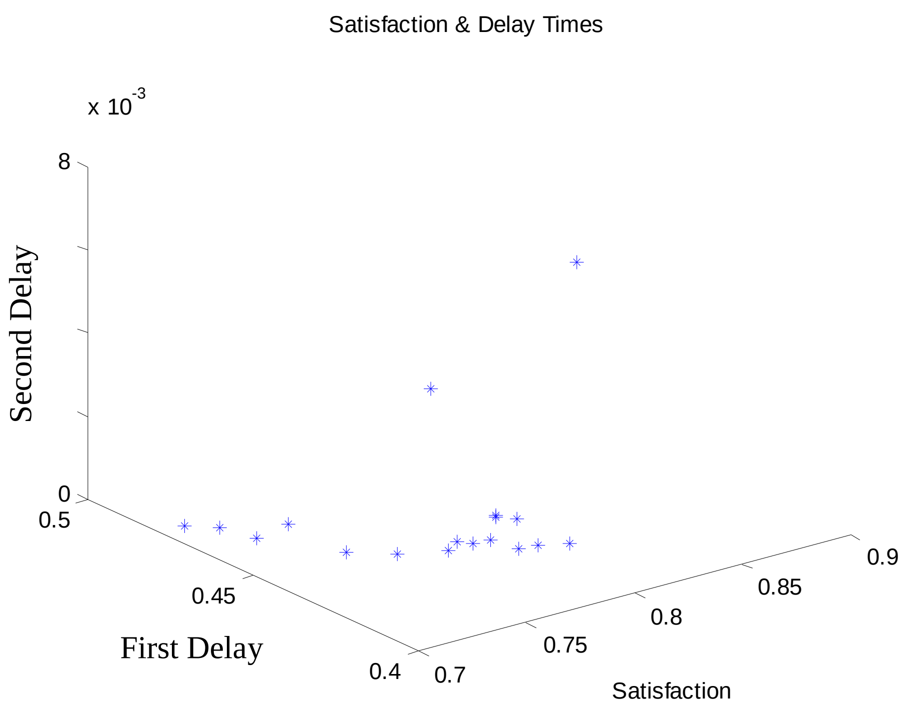

Figure 5.38. Satisfaction grade and both delays, population size 1000, file C109

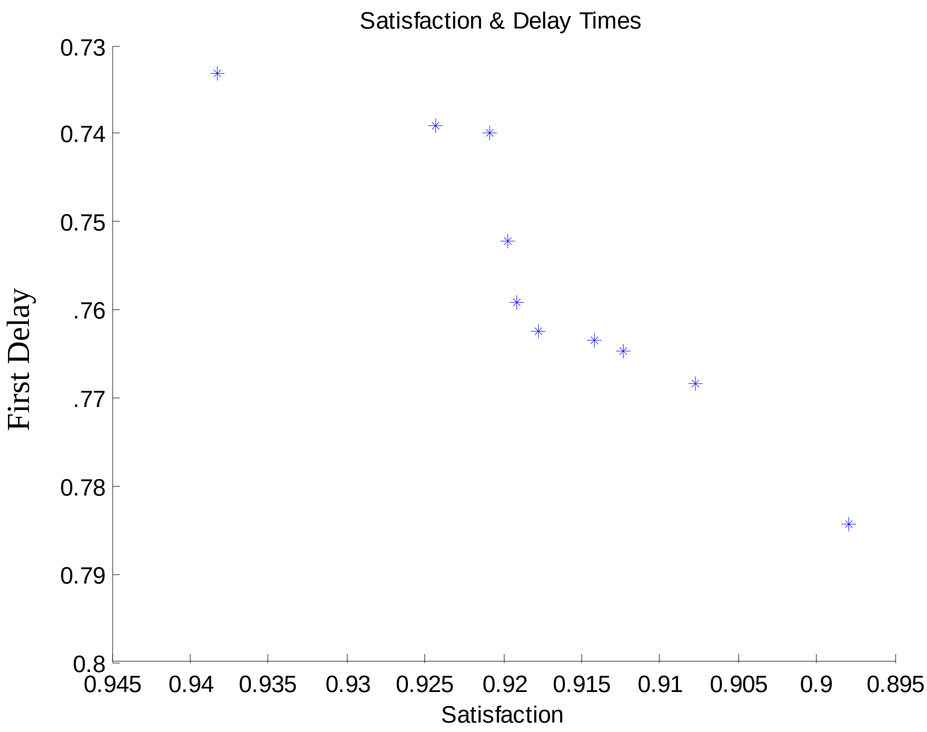

Figure 5.39. Satisfaction grade and both delays, population size 1000, file C208

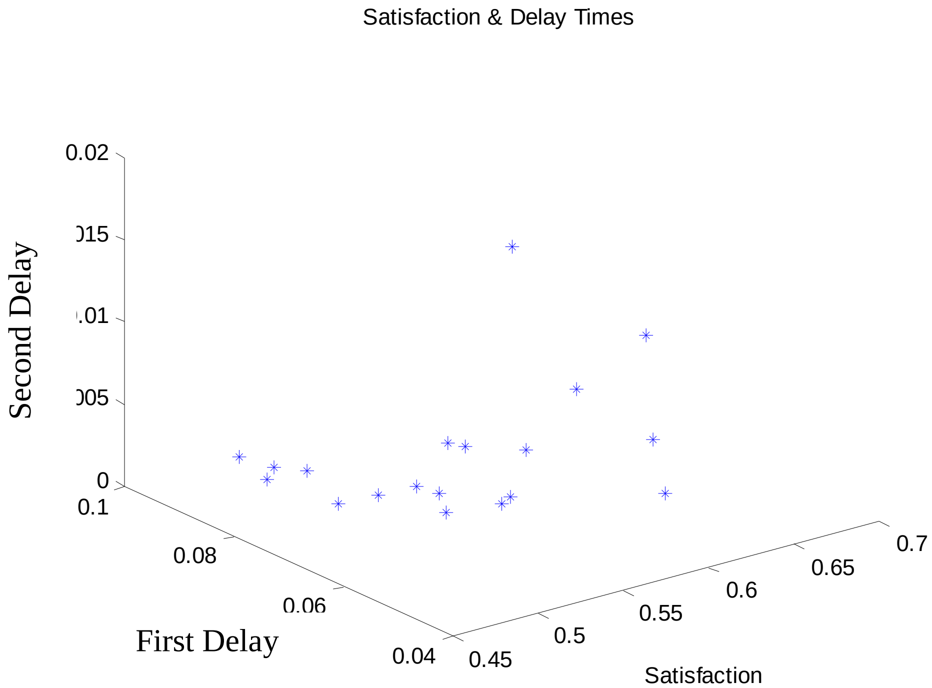

Figure 5.40. Satisfaction grade and both delays, population size 1000, file R112

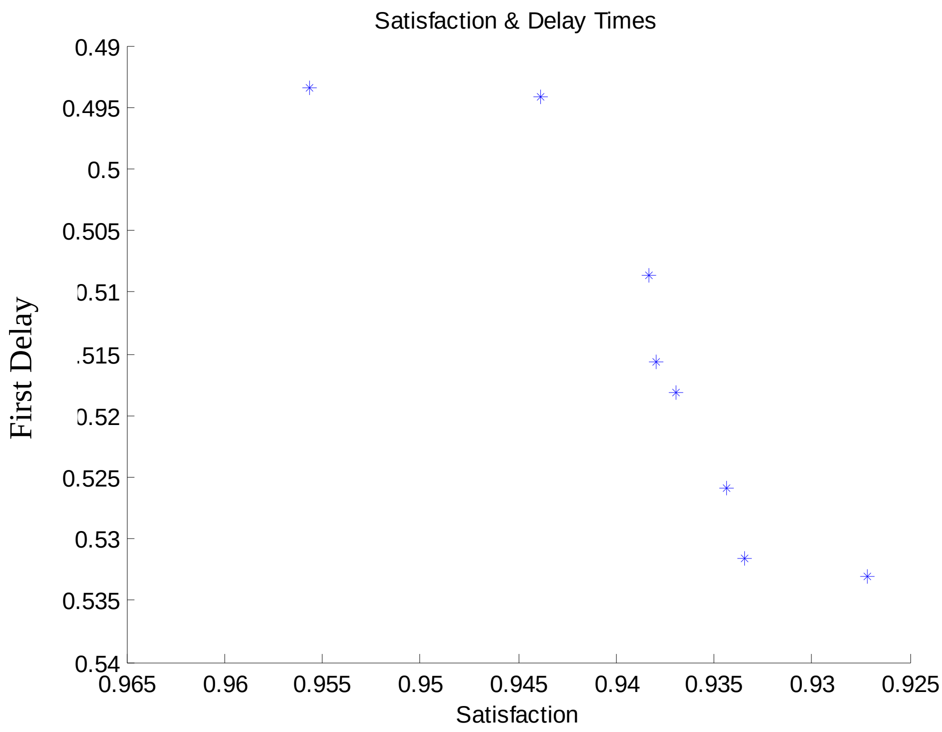

Figure 5.41. Satisfaction grade and both delays, population size 1000, file R211

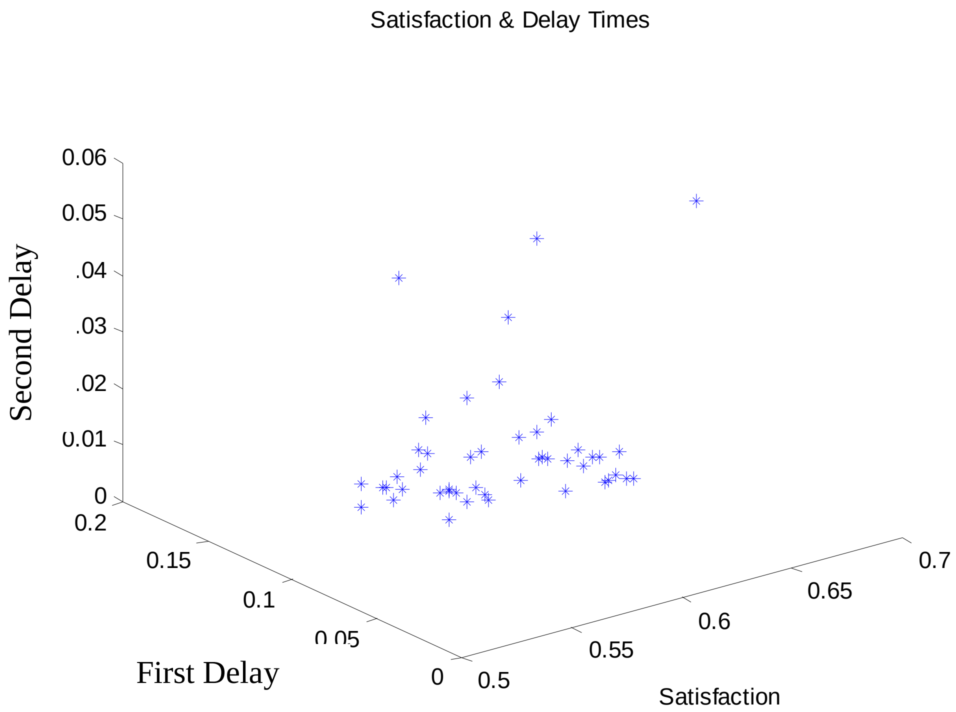

Figure 5.42. Satisfaction grade and both delays, population size 1000, file RC108

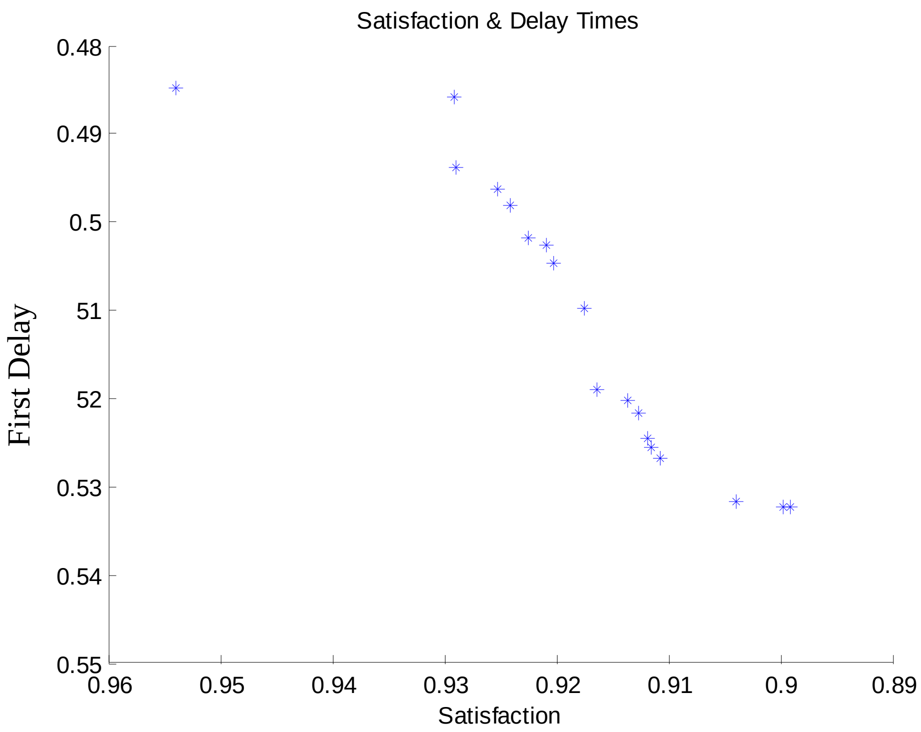

Figure 5.43. Satisfaction grade and both delays, population size 1000, file RC208

5.5 Satisfaction Grade, Waiting Time and Distance Travelled

The following experimental results are for other objective functions, which were also implemented in this thesis. These objective functions were implemented with population size 100 for all types of problems as shown in the following figures, where f(1) states for satisfaction grade objective function, f(2) is for waiting time and f(3) is for distance travelled.

As we can see from figures below, the waiting time of customers is in the range of 0.5-0.8 in average. However, in our proposed solution the objective function first delay, which is a waiting time of vehicles, is in range of 0.35-0.5. Therefore, the improvement is about 25%, so our approach and calculation of an objective function is better than the previous objective function by 25%.

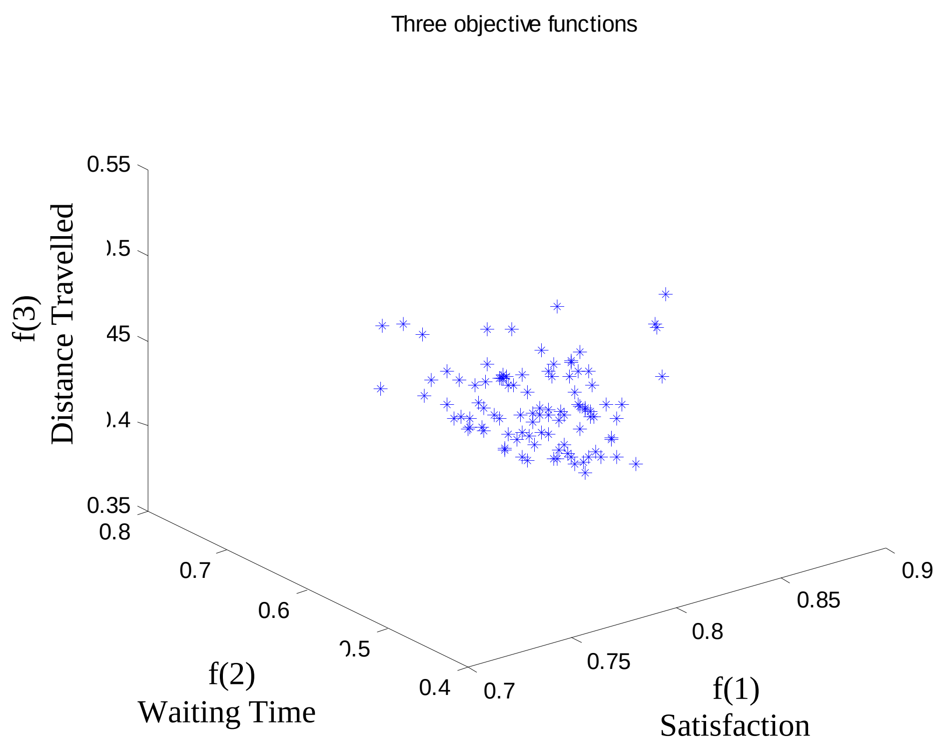

Figure 5.44. Three objectives, population size 100, file C109

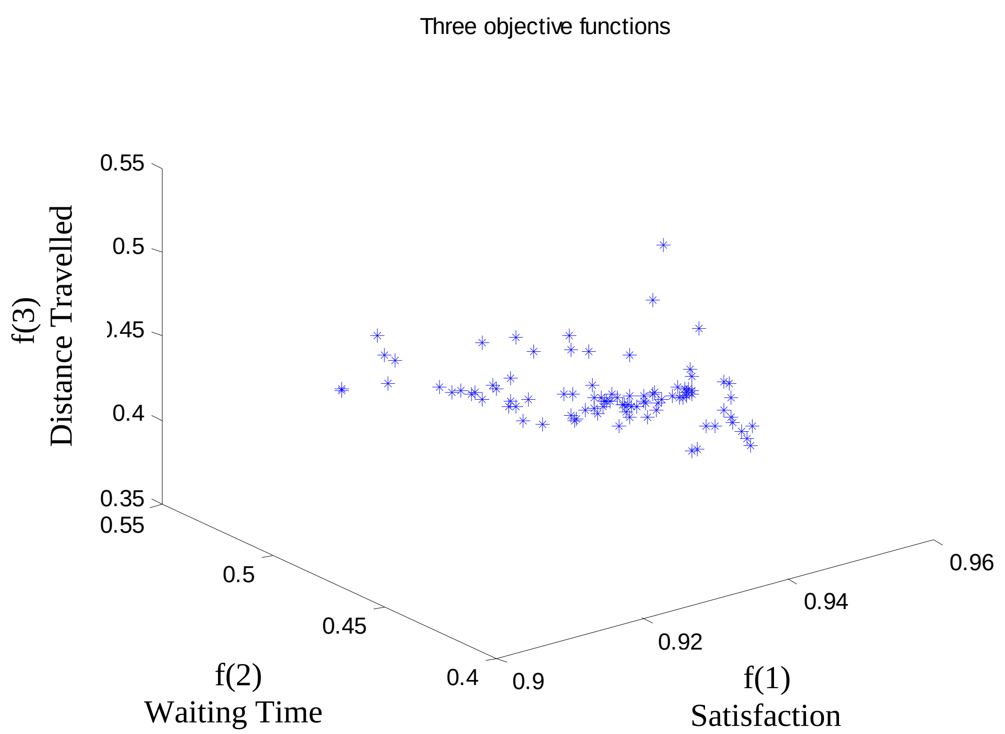

Figure 5.45. Three objectives, population size 100, file C208

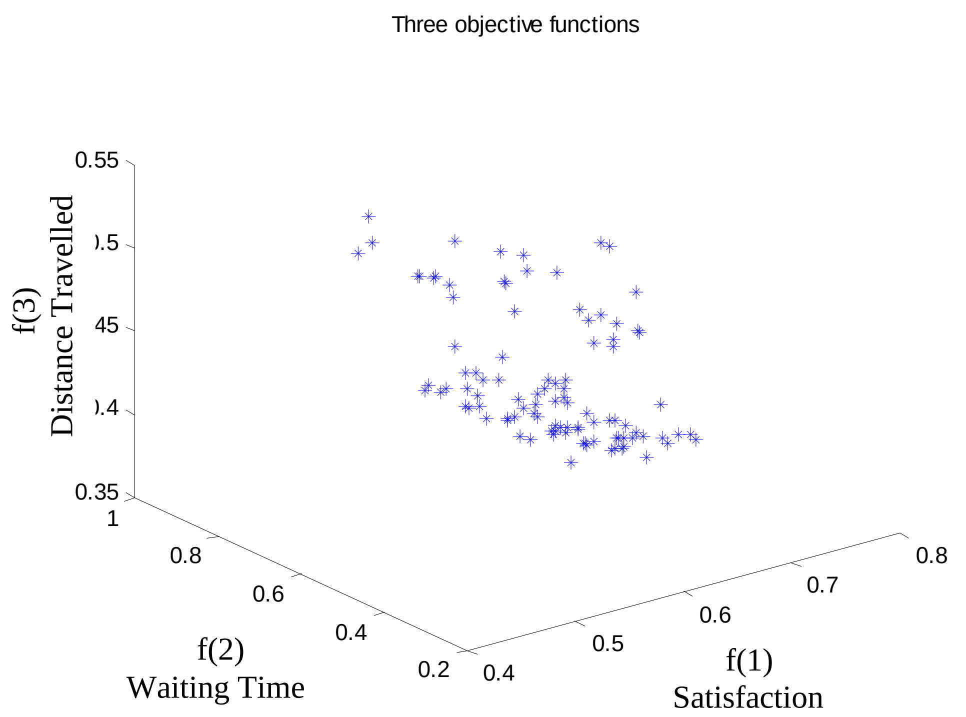

Figure 5.46. Three objectives, population size 100, file R112

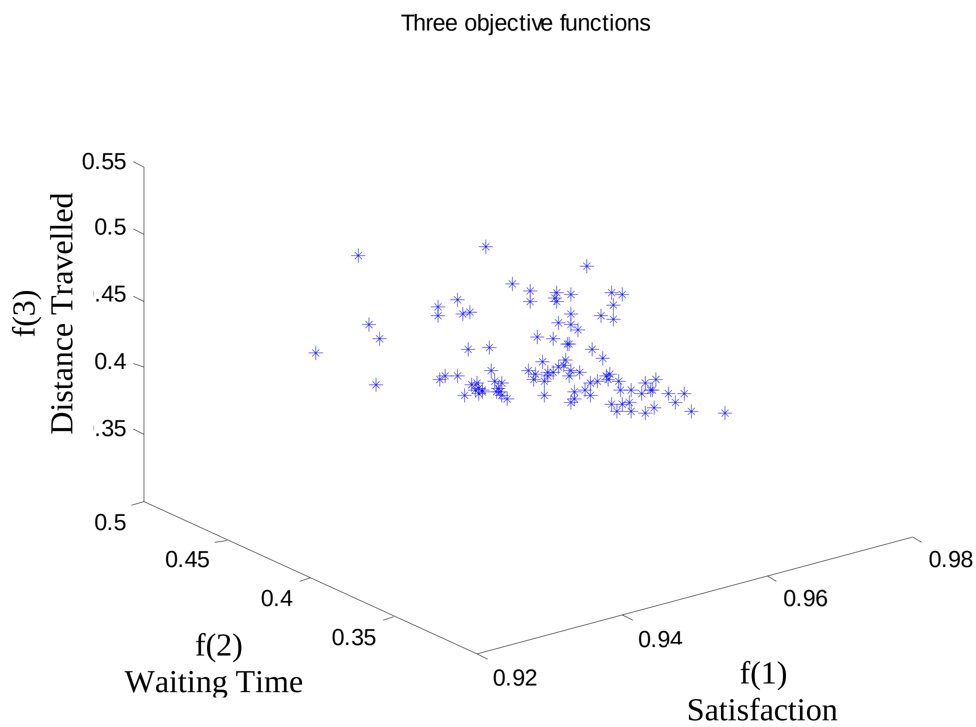

Figure 5.47. Three objectives, population size 100, file R211

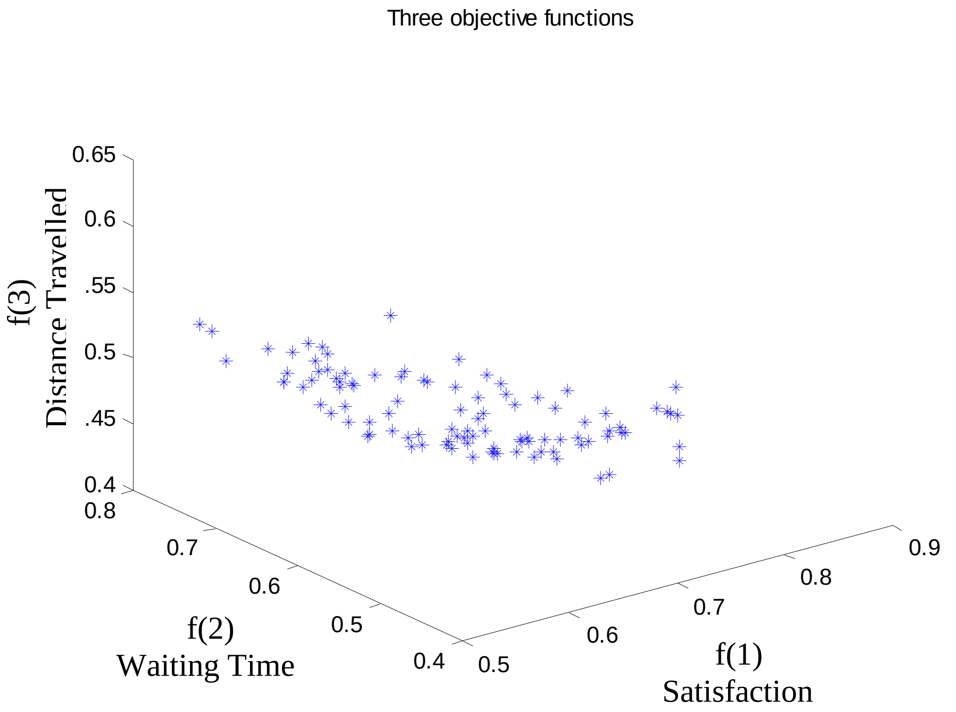

Figure 5.48. Three objectives, population size 100, file RC108

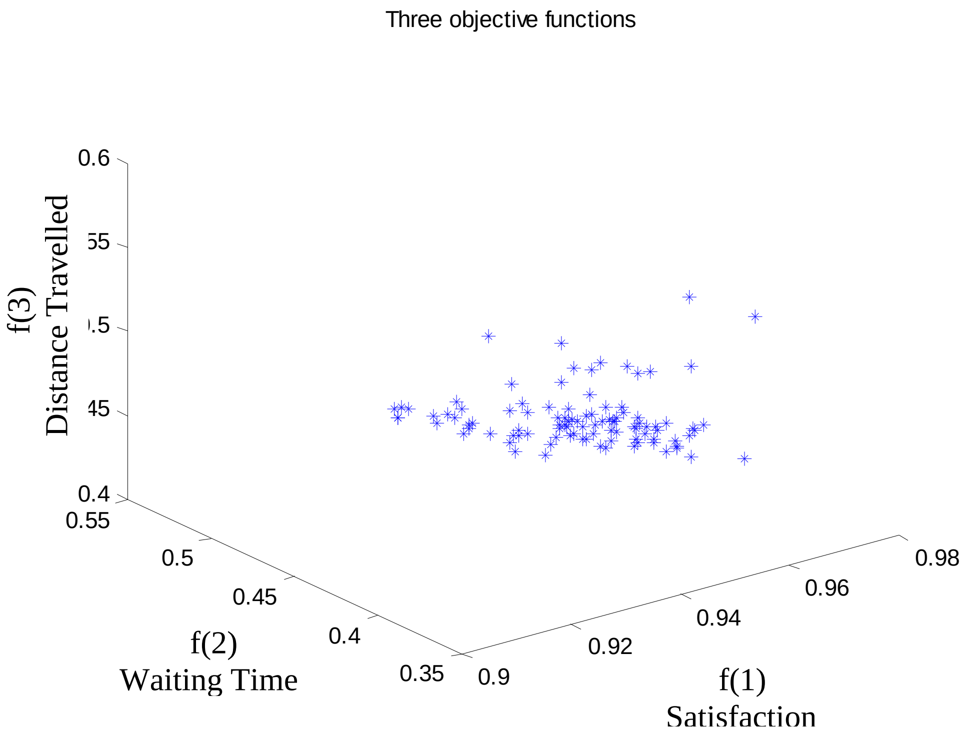

Figure 5.49. Three objectives, population size 100, file RC208

The results given above for three objectives show that as waiting time decreases the satisfaction grade of customers increases. However, the distance travelled does not affect waiting time or satisfaction grade of customers in terms of a constant increase or decrease. So, we cannot say that if distance travelled decreases, then waiting time also decreases or that satisfaction grade increases, because sometimes it might be just opposite.

In Table 5.1, our approach, using the delay time of vehicles, and the waiting time of previously proposed approach [26] are compared with respect to a waiting time of vehicles, since in our case the first delay is the waiting time of vehicles. In the previous proposed approach [26] the pure Genetic Algorithm was used as a search mechanism, in addition they used weight values, which were set to give more importance to some objective values with comparison to others. Moreover, the service satisfaction is compared with respect to the same approach in [26]. Comparison results are given for all six types of problems with population size 100.

Objective / ***Prob. type***	**Waiting time of vehicles**	**Satisfaction grade**	**First delay**	**Satisfaction grade**
C109	0.45 – 0.63	0.73 – 0.83	0.43 – 0.52	0.73 – 0.82
C208	0.44 – 0.51	0.91 – 0.96	0.74 – .78	0.905 – 0.94
R112	0.4 – 0.75	0.5 – 0.77	0.06 – 0.11	0.51 – 0.68
R211	0.34 – 0.49	0.925 – 0.975	0.48 – 0.55	0.935 – 0.96
RC108	0.4 – 0.8	0.54 – 0.8	0.06 – 0.18	0.54 – 0.69
RC208	0.39 – 0.51	0.92 – 0.975	0.49 – 0.54	0.9 – 0.96

Table 5.1. Comparison of objective functions results

As shown in the table above, the left hand side of a table with waiting time of vehicles and satisfaction grade is a previously proposed approach, whereas on the right hand side – first delay and satisfaction grade is a part of an approach proposed in this thesis, since there is also a second delay, which is the waiting time of customers. However, since waiting time of customers was not proposed before, there is nothing to compare with. Therefore, this objective function is not included in this comparison table.

Chapter 6

CONCLUSION

This work presents the solution approach and implementation of non-dominated sorting genetic algorithm for fuzzy vehicle routing problem. Multi-objective optimization approach of used algorithm reflects the solution to the problem, which has several conflicting objective functions to be optimized. Non-dominated sorting genetic algorithm is a multi-objective version of a classical genetic algorithm with better search methods, including the front-based dominance and a new crowding distance parameter.

In this fuzzy vehicle routing problem main concentration was on three objectives, which are minimization of waiting time of vehicles and customers and maximization of service satisfaction of customers. Obtained results from benchmark problem instances classified by geographical distribution of customers showed the significant difference and especially in minimizing the waiting time of vehicles objective function with comparison to proposed solution approach by other researchers.

REFERENCES

[1] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, A Fast Elitist Multiobjective Genetic Algorithm: NSGA-II, *IEEE Transactions on Evolutionary Computation* 6 (2002), no. 2, 182 – 197.

[2] G.B Dantzig and R.H. Ramser, The Truck Dispatching Problem, *Management Science* 6, 80-91, 1959.

[3] C. Zhang, Z. Zhang, L. Zheng and L. Cai, The Assignment of Customers in Multi-depot Vehicle Routing Problem with Fleet Size Constraint for Each Depot, *Automation and Logistics, 2007 IEEE International Conference*, 1897, 2007.

[4] L. Wen and F. Meng, An Improved PSO for the Multi-Depot Vehicle Routing Problem with Time Windows, *Computational Intelligence and Industrial Application, 2008. PACIIA '08. Pacific-Asia Workshop*, 852, 2008.

[5] Michael Wasner and Gunther Zapfel, An integrated multi-depot hub-location vehicle outing model for network planning of parcel service, *Int. J Production Economics* 90 (2004) 403-419.

[6] Wu TH, Low C and Bai JW, Heuristic solutions to multi-depot location-routing problems, *Computers and operations Research*, 2002, 29 (10), 1393-1415.

[7] Hirota K, Chen KW and Dong FY, Computational intelligence approach to real-world cooperative vehicle dispatching problem, *Second IEEE International Conference on Intelligent Systems*, June 2004, 7-12.

[8] W. Xuefeng, An Integrated Multi-depot Location- inventory-routing Problem for Logistics Distribution System Planning of a Chain Enterprise, *Logistics Systems and Intelligent Management, 2010 International Conference*, 1427, 2010.

[9] Fung, R.Y.K., J. Tang, J. Zhang, A Multi-Depot Vehicle Routing Problem with Weight-Related Costs, *Computers & Industrial Engineering, 2009. CIE 2009. International Conference*, 1028, 2009.

[10] Salhi, S. and M. Sari, A multi-level composite heuristic for the multi-depot vehicle fleet mix problem, *European Journal of Operational Research*, Vol. 103, No.1, pp95-112, 1997.

[11] Dondo, R. and 1. 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, No.3, ppI478-1507, 2007.

[12] Golden, B.L. and A.A. Assad (1986), “Perspectives on Vehicle Routing: Exciting New Developments”, *Operations Research* 34, 803−809.

[13] M. Desrochers, J. K. Lenstra, M. W. S. Savelsbergh and F. Soumis, *Vehicle Routing with Time Windows: Optimization and Approxinmation*, In “B. Golden and

A. Assad (eds.) Vehicle Routing: Methods and Studies, Amsterdam: Elsevier Science Publishers", 1988.

[14] M. M. Soloman and J. Desrosiers, *Time Window Constrained Routing and Scheduling Problems*, Transportation Science, 22 (1988), 1-13.

[15] Li, H., Lim, A., “”Local search with annealing-like restarts to solve the VRPTW”, *European Journal of Operational Research*, Corrected Proof, Article in Press., 2003.

[16] O. Arbelaitz, C. Rodríguez, I. Zamakola, “Low Cost Parallel Solutions for the VRPTW Optimization Problem”. *ICPP Workshops* 2001: 176-181.

[17] Y. Zhong and X. Pan, “A Hybrid Optimization Solution to VRPTW Based on Simulated Annealing,” *Proceedings of the IEEE International Conference on Automation and Logistics*, IEEE, Jinan, China, August 18 - 21, 2007.

[18] Taillard, E., Badeau, P., Gendreau, M., Guertin, F. and Potvin, J.-Y.. “A tabu search heuristic for the vehicle routing problem with soft time windows”. *Technical report* CRT-95-66, 1996.

[19] M. A. Cruz-Chávez, O. Díaz-Parra, J. A. Hernández, J. C. Zavala-Díaz, M. G. Martínez-Rangel. “Search Algorithm for the Constraint Satisfaction Problem of VRPTW”. *Electronics, Robotics and Automotive Mechanics Conference*, CERMA2007, IEEE-Computer Society, ISBN 0-7695-2974-7, pp 746-751, 25-28 September, México, 2007.

[20] Garcia, B.L., J.Y. Potvin and J.M. Rousseau. "A Parallel Implementation of the Tabu Search Heuristic for Vehicle Routing Problems with Time Window Constraints", *Computers & Operations Research*, 21(9):1025–1033, 1994.

[21] C. Qi, Y. Sun. “An Improved Ant Colony Algorithm for VRPTW”. CSSE (1), 455-458, 2008.

[22] C. Qi, S. Cui, Y. Sun. “Using Ant Colony System and Local Search Methods to Solve VRPTW”. PACIIA (2), 478-482, 2008.

[23] Thangiah SR, Nygard PL, Juell PL. “A genetic algorithm system for vehicle routing with time windows”. *In Proceedings of the Seventh IEEE Conference on Arti2cial Intelligence Applications*, Miami, FL, p. 322-328, 1991.

[24] Braysy, O., Berger, J., Barkaoi, M., “A new hybrid evolutionary algorithm for the vehicle routing problem with time windows”. Presented at the *Route 2000-Workshop,* Skodsborg, Denmark, 2000.

[25] Runwei C, Mitsuo G, Tatsumi T. “Vehicle routing problem with fuzzy due-time using genetic algorithms”. *Journal of Japan Society for Fuzzy Theory and Systems* 7(5):1050–1061, 1995.

[26] J.J. Lin, “Multi-Objective Decision Making for Vehicle Routing Problem with Fuzzy Due Time”, *2006 IEEE International Conference on Systems, Man, and Cybernetics*, Vol.4, pp. 2903-2908, 2006.

[27] R. Gupta, B. Singh and D. Pandey, “Fuzzy Vehicle Routing Problem with Uncertainty in Service Time”, *International Journal Of Contemporary Mathematical Sciences*, Vol.5, No.11, pp. 497-507, 2009.

[28] R. Gupta, B. Singh and D. Pandey, “A Genetic Approach for Fuzzy Vehicle Routing Problems with Time Wondows, *communicated to Indian Academy of Mathematics*, Indore, India, 2009.

[29] R. Gupta, B. Singh and D. Pandey, “Multi-Objective Fuzzy Vehicle Routing Problem: A Case Study”, *International Journal Of Contemporary Mathematical Sciences*, Vol.5, No.29, pp. 1439-1454, 2010.

[30] H. Beyer and K. Deb. On Self-Adaptive Features in Real Parameter Evolutionary Algorithm. *IEEE Trabsactions on Evolutionary Computation*, 5(3):250-270, 2001.

[31] K. Deb and R. B. Agarwal. Simulated Binary Crossover for Continuous Search Space. *Complex Systems*, 9:115-148, 1995.

[32] M. M. Raghuwanshi and O. G. Kakde. Survey on multi-objective evolutionary and real coded genetic algorithms. In *Proceedings of the 8th Asia Pacific Symposium on Intelligent and Evolutionary Systems*, pages 150-161, 2004.

[33] M. Pinedo, Scheduling Theory, *Algorithms, and Systems*, ISBN: 0130281387, 2 Ed. Prentice Hall, U.S.A., Aug. 2001.

[34] M. Reimann, K. Doerner and R.F. Hartl. Analyzing a Unified Ant System for the VRP and Some of Its Variants. *EvoWorkshops 2003*, LNCS 2611, pp. 300–310, 2003.

[35] Polat, L., Acan, A., Unveren, A. Cooperative coevolutionary algorithms for fuzzy vehicular routing problem: An analysis of efficiency vs. geographical distribution. *Evolutionary Computation, 2007. CEC 2007. IEEE Congress*, pp. 1126, 2007.

[36] (Accepted paper) Zhavat S., Unveren A. and Acan, A. An Evolutionary Multi-Objective Modeling and Solution Approach for Fuzzy Vehicle Routing Problem. *INISTA 2011, IEEE Conference.*