A TABU SEARCH HEURISTIC FOR THE VEHICLE ROUTING PROBLEM WITH TIME DEADLINES AND ASYMMETRIC DISTANCES

by

Pelin Ekmen

B.S., Industrial Engineering, Yıldız Technical University, 2010

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering

Boğaziçi University

2013

to my family

ACKNOWLEDGEMENTS

Foremost, I would like to thank my thesis supervisor Necati Aras for his orientation, guidance, understanding, and encouragement throughout this study. This study would not have been possible without his support and his tolerance.

I would like to thank my co-supervisor Deniz Aksen for his advisory, support, guidance and encouragement during this thesis.

I want to thank my thesis committee members Kuban Altınel, Tınaz Ekim Aşıcı and F. Sibel Salman for taking time to examine this thesis and taking part in my thesis jury.

My deepest gratitude goes to my family for their unflagging love, patience and support throughout my life.

I would especially like to thank Selçuk Güler and Ahmet Açıkyol for their help and support in encoding. Without them, I would never learn C# programming language so quickly.

I wish to express special acknowledgements to Gizem for being always there to listen and help me since my B.S., to Yasemin and Özlem, who are also my neighbors, for kindly offering their never ending help, emotional support and patience during this research even in hardest times, to Özge for her sincere help and inspiring motivation, and to Kadir for his resourcefulness, willingness and funny stories. I did not expect to have such great friendships in graduate school before I started my master. My time here was much more colorful and encouraging than I could ever think of.

I would also like to thank TÜBİTAK for providing scholarship during my master program.

Furthermore, I would like to emphasis that it is a great pleasure for me to make my thesis in Boğaziçi University and to be a part of this family.

ABSTRACT

A TABU SEARCH HEURISTIC FOR THE VEHICLE ROUTING PROBLEM WITH TIME DEADLINES AND ASYMMETRIC DISTANCES

Reducing transportation cost and meeting customer requirements on time are the two main targets of transportation, logistics, distribution and supply chain management. In this thesis, we focus on the Vehicle Routing Problem with Time Deadlines and Asymmetric Distances which appears as an application in transporting cash to ATMs. Two mathematical models are presented to solve this problem. Since large instances cannot be solved exactly, a metaheuristic method based on Tabu Search algorithm is also proposed to determine near-optimal routes by formulating a mixed integer linear model.The algorithm is tested on randomly generated asymmetric instances derived from the Solomon benchmark problem instances. They are solved by CPLEX 12.5 solver within GAMS suite 24.0 and compared with the results obtained by the Tabu Search heuristic. Furthermore, we convert some of Christofides *et al*. classical VRP test problems known as CMT in the literature available, and solve them by our algorithm. We compare our solutions with the best known results in the literature.

ÖZET

ZAMAN SINIRLI, İKİ YÖN MESAFELİ ARAÇ GÜZERGAH BELİRLEME PROBLEMİ İÇİN YEREL ARAMALI BİR ÇÖZÜM YÖNTEMİ

Tasıma maliyetlerini azaltmak ve müşteri ihtiyaçlarını zamanında karşılamak ulaşım, lojistik, dağıtım ve tedarik zinciri yönetiminin iki önemli hedefidir. Bu tezde çalışmasında, ATM'lere para nakliyesindeki bir uygulamada görülen İki Yön Mesafeli, Zaman Kısıtlı Gerçek Hayat Araç Güzergah Belirleme Problemi üzerinde çalışılmıştır. Problemin iki matematik modeli sunulmuştur. Büyük örnekler tam olarak çözülemediği için, tam sayılı doğrusal model hazırlanarak, Yerel Aramalı çözüm yöntemine dayanan bir en iyileme metodu ile en iyiye yakın güzergahların belirlenmesine çalışılmıştır. Solomon'un karşılaştırma problem örnekleri yardımıyla rassal olarak iki yön mesafeli örnekler üretilmiş ve çözüm yolu test edilmiştir. Bu örnekler ayrıca GAMS 24.0 aracılığıyla CPLEX 12.5 çözücüde çözülmüş ve Yerel Aramalı yöntemle elde edilen çözümlerle karşılaştırılmıştır. Ayrıca literatürde CMT olarak bilinen Christofides ve arkadaşlarının klasik araç güzergah belirleme test problemlerinden bazılarını bizim problemimize çevirdik ve kendi çözüm yolumuzla çözüp sonuçları literatürdeki bilinen en iyi sonuçlarla karşılaştırdık.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

a_i	Required demand for customer i
$\overline{a_i}$	Average demand used to normalized
c	Travelling cost per unit distance
d_{ij}	Asymmetric travelling distance between customers i and j
e_i	Earliest arrival time of customer i
F_{ij}	Amount of flow from customer i to j
I	Set of customers and depot
IC	Set of customers only
i, j	Index for customers
K	Number of vehicle obtained in a solution
k	Index for vehicles
l_i	Latest arrival of customer i
M	A large number
max$Incum_{iter}$	Maximum number of iterations through which incumbent solution does not improve
$max\ _{iter}$	Maximum number of iterations
n	number of customer
num_i	Total number of times move i is used so far in previous iteration
$prob_i$	Probability of choosing move i
q	Capacity of vehicles
R_i	Arrival time at customer i
R_{ik}	Arrival time at customer i by vehicle k
s	A solution for the problem
s_i	Service time for customer i
S_{ij}	Savings for pairs of customers i and j
t_{ij}	Asymmetric travelling time between customers i and j
X_{ij}	Binary variable showing whether distance between customers i and j traversed by a vehicle

X_{ijk}	Binary variable showing whether distance between customers i and j traversed by vehicle k
z	A solution value
α	Penalty factor for over capacity
β	Penalty factor for over tour duration and latest arrival time
η	Parameter for considering customer demands while calculating savings
φ	Parameter for considering asymmetric distances while calculating savings
λ	Route shape parameter
μ	Sparsification parameter
δ	Penalty parameter for updating penalty factors α and β
θ	Random value for creating asymmetric distances
ν	Granularity threshold value

LIST OF ACRONYMS/ABBREVIATIONS

BC	Bi-criteria
CW	Clarke and Wright
ENS	Expanding neighborhood search
GA	Genetic algorithm
GRASP	Greedy randomized adaptive search procedure
HFVROTW	Heterogeneous fleet vehicle routing with overloads and time windows
HVRP	Heterogeneous fleet VRP
LP	Linear program
LR	Lagrangian relaxation
LS	Local search
MCPSO	Multi swarm cooperative particle swarm optimization
MDPVRPTW	Periodic and multi depot VRPTW
MDVRP	Multi depot VRP
MDVRPTW	Multi depot VRPTW
MDVSPTW	Multi depot vehicle scheduling problem with time windows
MPVRP	Multi period VRP
OEM	Original equipment manufacturer
OVRP	Open VRP
PSO	Particle swarm optimization
PVRPTW	Periodic VRPTW
REVLOG	European working group on reverse logistics
SDVRPTW	Site-depended VRPTW
SS	Scatter search
TS	Tabu search
VCPS	Vehicle capacity planning system
VRP	Vehicle routing problem
VRPB	VRP with backhauls
VRPHTW	Vehicle routing problem with hard time windows
VRPPD	VRP with pick-up and delivery

VRPSTW	Vehicle routing problem with soft time windows
VRPTD	Vehicle routing problem with time deadliness
VRPTW	Vehicle routing problem with time windows
VRPTWSD	Vehicle routing problem with time windows and split deliveries

1. INTRODUCTION

The optimal distribution of products and the optimal delivery of services to customers are the main concerns of any industry involved in logistics management. Besides reducing the huge amounts of money spent on transportation, meeting customers' requirements on time becomes the main target of companies. Thus, they pay much attention to improving the efficiency of their logistics and distribution operations. This can be achieved by designing effective and efficient vehicle routes for customer service. Even though it is hard to incorporate all characteristics of distribution problems to the models due to the real life complexities, Vehicle Routing Problem with Time Windows (VRPTW) can handle most of them.

The VRPTW is a generalization of the VRP where every customer i must be visited within a given time windows $[e_i, l_i]$. It deals with the situation, where each of a number of identical vehicles departs from a depot, visits a set of customers to satisfy their demand and returns to the same depot by obeying the vehicle capacity limitation and time windows associated with each customer. The Vehicle Routing Problem with Time Deadlines (VRPTD) is a special VRPTW where the earliest arrival time is relaxed. The time deadline designates the latest time until which a vehicle should begin service at the customer. There may exist maximum tour duration associated with each vehicle, which restricts each vehicle to complete its tour. Obviously, the demand of each customer must be satisfied by exactly one vehicle.

VRPTW is an NP-hard problem, which is also true for VRPTD. Although there are exact solution methods, they cannot deal with large instances within a reasonable amount of time. Researchers develop metaheuristics for this difficult problem to generate near-optimal solutions.

In this thesis, we consider asymmetric distances among customers. The objective function is the minimization of the traveling time of the employed vehicles. We use the Tabu Search algorithm as a solution approach. Since the quality of the initial solution may have an effect on the overall solution, Clarke & Wright (CW) Savings algorithm is utilized

before Tabu Search algorithm. In order to improve our algorithm with respect to computation time, we define granular neighborhoods. Seven types of neighborhood structures are defined and a local search algorithm is applied to initial solution obtained by CW algorithm, to current solution at 100^{th} each iteration, and the best feasible solution whenever it is updated.

We apply our solution method to a real-world problem encountered by a security carrier company in Turkey. The company carries cash money from a central depot to ATMs by homogeneous vehicles. Each ATM has a specific demand, and has to be satisfied once by exactly one vehicle in a limited time. This problem is a typical real-life application of VRPTD. In most VRPTD, the travelling distances are assumed symmetric calculated by the Euclidean metric. However, in this thesis, we consider asymmetric distances.

The remainder of this thesis is organized as follows. Chapter 2 provides a comprehensive literature review related to the VRPTW and VRPTD problems. In Chapter 3 the mathematical model and its components are presented. The Tabu Search heuristic method for VRPTD with asymmetric distances is described in Chapter 4. Finally, Chapter 5 provides computational results of proposed algorithm.

2. LITERATURE REVIEW

2.1. Vehicle Routing Problem

The Vehicle Routing Problem (VRP), introduced by Dantzig and Ramser in 1959 [1], is a class of problems that determine the best tour for each vehicle in the fleet with optimum assignment such that each customer is visited by one vehicle only. Vehicles start from a depot and after visiting a set of customers, they return to the depot. VRPs have many real life applications in fields of transportation, logistics, distribution and supply chain management.

The most widely used objective function of VRPs is the minimization of the total tour length.

VRP is an important and well-known combinatorial optimization problem, which is shown to be NP-hard [2].Therefore, exact solutions cannot be found for large instances. Hence, heuristics and metaheuristics are commonly developed by researchers.

While hundreds of papers have been written to find exact and heuristic solutions for VRPs, it still attracts the attention of many researchers. Thus, more complicated metaheuristics are developed to obtain higher quality solutions for the problem.

There are different variants of VRP. These are VRP with Time Windows (VRPTW), VRP with Pick-up and Delivery (VRPPD), VRP with Backhauls (VRPB), Multi Depot VRP (MDVRP), Multi-Period VRP (MPVRP), Heterogeneous Fleet VRP (HVRP) and Open VRP (OVRP). Marinakis and Migdalas [3] present an annotated bibliography for the VRP.

2.2. Vehicle Routing Problem with Time Windows

The VRPTW is a generalization of the VRP where every customer i must be visited within a given time windows $[e_i, l_i]$. There are two versions: VRP with hard time windows

(VRPHTW) and VRP with soft time windows (VRPSTW). While customers can be served any time in VRPSTW by adding a time window violation penalty cost, arriving at a customer after the latest time or before the earliest time is not allowed for VRPHTW.

Early solution methods proposed for VRPTW cover simple heuristics or metaheuristics, but more complicated heuristics and metaheuristics have been developed over the time to obtain better results.

In the early studies, Solomon [4] analyze tour building algorithms and provide test problem sets for VRPTW. Ahn and Shin [5] are interested in VRPTW with time varying traffic congestion and solve it by insertion, savings and arc exchange heuristics, Desrochers *et al*. [6] solve the LP relaxation of the set partitioning formulation of the VRPTW by column generation. Thangiah [7] propose genetic algorithms consisting of a global clustering method and a local post-optimized method. Antes and Derigs [8] present a classical insertion principle with the negotiation between customers and tours.

Potvin *et al.* [9] describe a Tabu Search (TS) algorithm for VRPTW and compare it with other heuristics on 56 test problems of Solomon. With respect to the total number of vehicles and total route time, better solutions are achieved than Solomon's solutions and the results are competitive with other heuristics. In the second part of their work, genetic search algorithm is proposed and the generated solutions are competitive with the best known solutions [10].

Desaulniers *et al.* [11] consider multi-depot vehicle scheduling problem with time windows (MDVSPTW) and formulate it as an integer nonlinear programming problem based on multi commodity flows in graph theory. A column generation approach embedded in a branch and bound framework is used to handle the problem. With the travel times and vehicle fixed cost, the waiting times between two consecutive tasks is also considered as a cost, and they are minimized. Testing results indicate that the solution approach is very efficient in terms of optimal and heuristic versions. High quality solutions can be found in a reasonable amount of time by this approach.

Liu and Shen [12] use several insertion based savings heuristics for solving VRP with a heterogeneous fleet. The method is based on the traditional insertion aspect, which provides a better solution quality.

Cordone and Calvo [13] develop a heuristic for VRPHTW. To find an initial solution, they use an alternate K-exchange Reduction (AKRed) method that is based on a strategy combining k-exchange to escape local optima and an ad hoc procedure to reduce the number of vehicles. The developed algorithm does not include fixed parameters and random choice procedure. The results indicate that the algorithm is more effective than benchmark algorithms.

Lee *et al.* [14] consider vehicle capacity planning system (VCPS) as a VRPTW for real case study. The aim of the VCPS is to select job for outsourcing and minimize the total cost. To this end, TS algorithm is proposed and total cost is decreased up to 8.14%.

VRPTW with limited number of vehicles is considered by Lau *et al.* [15] To deal with this problem, they propose a TS algorithm. The violation of TW constraint is allowed with a penalty. An upper bound for the total number of customers that can be served by a given number of vehicles is obtained by an integer programming formulation. The results obtained by the algorithm are very close to the upper bound.

Cordeau *et al.* [16] are interested in periodic and multi-depot VRPTW (MDPVRPTW). They aim to provide an algorithm for the PVRPTW and the MDVRPTW since there is no algorithm reported for this generalized type of VRPTW. A simple, robust and efficient unified tabu search algorithm is proposed. They modify their algorithm for the PVRPTW, the MDVRPTW, and the site-dependent VRPTW (SDVRPTW) in another study in 2004. The new improved algorithm also deals with the route duration constraint and the obtained results improve or match the previous best-known solutions.

Bräysy *et al.* [18] suggest a multi start local search heuristic based on a two-phase approach for solving the VRPTW. In the first phase, multiple initial solutions are generated by using a constructive heuristic. Injection trees (IT) which is an extension of the well-known ejection chain (EC) approach, are applied for reducing the number of routes. In the

second phase, improvement heuristics based on cross exchange is used to minimize intra- and inter-tour distances and a threshold accepting (TA) post-processor, the solution is improved.

Berger *et al.* [19] develop a route directed hybrid genetic approach for VRPTW with a fixed number of tour. Populations are classified into two groups with separate objectives: minimizing the total travelling distance and minimizing temporal constraint violation. With this method, some new best-known solutions are obtained. Lather, Berger and Barkaoui [20] propose a parallel hybrid genetic algorithm with the same procedure. As the parallel procedure, they used a master-slave message-passing paradigm with the aim of yielding additional speed-up. The proposed method is very competitive with the best-known heuristics and it provides several best-known results.

Homberger and Gehring [21] design a two-phase hybrid metaheuristic with the aim of minimizing the number of vehicles by using an evolution strategy in the first phase and minimizing the total distance by using a TS algorithm in the second phase. 360 problems from the literature are tested and to obtained better solutions.

Haghani and Jung [22] study pick-up and delivery VRPSTW. Heterogeneous vehicles, real time service requests and travel time variations are also included to their study. They present genetic algorithm to solve problem. With up to 10 demand points, their solutions give same result as exact solutions but better computational time is required. For the larger problem, acceptable solutions are handed. Their algorithm gives better solutions when unanticipated nature of problem decreases.

Montemanni and Gambardella [23] adopt an ant colony system for dynamic VRPTW in which vehicles gets information about the next customer assigned to them during their travel, and therefore they do not have to go back to the depot. The proposed method is tested in both artificial and real life data instances.

Ombuki *et al.* [24] consider the VRPTW as a multi objective optimization problem. Minimizing the total distance and minimizing the number of vehicle are taken into account

as separate measures by using Pareto ranking procedure. GA is used to cope with the problem and it yields competitive solutions.

Repoussis and Paraskevopoulos [25] develop a hybrid metaheuristic by using multi start local search approach. Greedy randomized adaptive search procedure and variable neighborhood TS are introduced to the method as a diversification and intensification mechanism, respectively. This method produces efficient and robust solutions for benchmark instances. In their following study [26], they tackled with the heterogeneous fleet VRPTW. A two-phase solution approach based on a hybridized TS and a new Reactive Neighborhood Search metaheuristic algorithm were employed with providing high quality solutions.

Kallehauge *et al.* [27] propose Lagrangian Relaxation (LR) method for VRPTW and then constrained shortest path sub problem is solved by stabilized cutting plane algorithm in branch and bound scheme. Compare to traditional column generation based algorithm on a large number of test problems, their method gives significant speedup.

Fu *et al.* [28] describe different types of time window violations and in order to solve the vehicle routing problem with soft time windows, they present a unified TS algorithm with unified penalty function. The capacity and maximum tour duration constraint are allowed by applying different penalty functions. By using the mixed neighborhood structure and randomly selected move operator in their TS algorithm, they reach better solutions compared to the best-known solutions for benchmark problem.

Dell'Amico and Monaci [29] determine at the same time, the composition and the routing of a fleet of heterogeneous vehicles. The fleet consists of different type of vehicles, and the vehicles have a fixed cost with a loading capacity. A constructive insertion heuristic and a metaheuristic algorithm are proposed to minimize vehicle fixed cost and variable routing costs.

Ostertag and Doerner [30] cope with a real life large scale multi depot VRPTW. POPMUSIC framework is developed for a decomposition approach and as an optimizer in

the POPMUSIC framework, a memetic algorithm was used. Proposed method is efficient and flexible in terms of solution quality and runtime.

Cheng and Wang [31] decompose the VRPTW into a clustering problem and a set of independent traveling salesman sub problems by a decomposition technique to reduce problem size and use simple solution method. For the clustering problem, genetic algorithm is used while a simple heuristic algorithm is presented for traveling salesman problems. Iterative interactions between the main problem and the set of sub problems consist of the original problem's solution. The presented algorithm provides better solutions compared with the insertion method.

Salani [32] describe an extension of VRPTW where customers request several discrete items. A classical constraint of the problem that each customer is visited only once by only one vehicle is changed to that a customer can be visited by more than one vehicle but each vehicle can only serve one type of customer order. With this respect, a mixed integer program is modeled for the problem and branch and bound algorithm is presented. The method is tested with a limited number of instances and the results are not efficient.

Ho and Haugland [33] study a case of VRPTW, VRPTW and split deliveries (VRPTWSD). In this problem, visiting a customer whose demand exceeds the vehicle capacity by more than one vehicle is allowed in order to cope with the split deliveries. They use TS algorithm for solving the VRPTWSD and as a neighboring structure, union of four moves operator is defined. They test how affects splitting option to the objective function. It is seen that this helps to decrease distance travel and the number of vehicle.

Belfiore *et al.* [34] are also interested in VRPTWSD. They propose a Scatter Search (SS) algorithm and as an initial solution the sequential insertion heuristic of Solomon [4] was used. The experimental results are better than the best results from [33].

Ma [35] applies an improved ant colony algorithm to solve VRPTW. Pheromones adjust strategies improves search results by avoiding to fall into local optimization while path construction strategy helps to improve the path length. Hence, an efficient optimized solution is handled with proposed method.

Mao and Deng [36] develop an hybrid evolution algorithm consisting of GA, greedy randomized adaptive search procedure (GRASP), the expanding neighborhood search (ENS) strategy and particle swarm optimization (PSO). PSO is employed for interacting two generations in order to make parents more competitive. GRAPS is used to find initial generation of particles and ENS is performed for improving the solutions. The suggested method is seen to reduce the total distances.

Carlos *et al.* [37] cope with a capacitated VRPTW (CVRPTW) with multi objectives that are minimizing the total number of vehicles, and minimizing the total travel time of the used vehicles. Mixed integer programming is used and it is tested in a real life problem of a distribution company. Good results are obtained in a reasonable time.

Potvin *et al.* [38] study on dynamic events that relate new customer request and dynamic travel times in vehicle routing and scheduling problem. They consider tolerance concept as reactive dispatching strategies instead of disregarding reaction or making immediately reaction. As a dispatching algorithm, insertion heuristic is used. To handle dynamic requests, simulation is used, and the same local improvement phase is repeated. With the introducing tolerance factor to dynamic request, they come out with better solutions. They extend their study by considering dynamic travel times that drivers' current position can be changed immediately when a new customer request is received [39].

Gan *et al.* [40] introduce uncertain number of vehicles with simultaneous delivery and pick up service into VRPTW. To minimize the transport cost as an objective function of the problem, they use multi swarm cooperative particle swarm optimization (MCPSO). The method is formed by the strategy that while master swarms alter their particles with their own knowledge and particles of the slave swarms' knowledge, the slave swarm perform PSO independently. Besides, they use a different encoding structure. That contains customer served order in a particular vehicle. The proposed algorithm performs better than the GA and PSO algorithm.

Gong *et al.* [41] use a set based PSO for the VRPTW by considering the problem space as a directed completed graph and considering each candidate solution as a spanning sub-graph of the completed graph. The particle is initialized by a nearest neighbor heuristic

(NNH) its velocity is adjusted with historical search experiments. Violation of the VRPTW constraints is not allowed. Proposed algorithm yields better solutions than best-known results for Solomon instances.

The limited number of vehicles causes overtime costs. This situation triggers to think about a tradeoff between the overtime and outsourcing cost. Moon *et al.* [42] are interested in VRPTW considering overtime and outsourcing vehicles. For handling with this problem, they develop a mixed integer programming, an efficient GA and an efficient hybrid algorithm based on simulated annealing. They also design a decision support system which yields a vehicle routing selection process and scheduling optimization.

Kritikos and Ioannou [43] present a sequential insertion heuristic for the heterogeneous fleet vehicle routing with overloads and time windows (HFVROTW). In addition to using vehicles with different capacities in the classical heterogeneous fleet VRP, they tackle vehicles which are allowed to be loaded above given capacities. The overload capacities are considered as a penalty cost for objective function.

Vidal *et al*. [44] develop a hybrid genetic search for a large scale VRPTW. They present new evaluation methods including infeasible solutions with respect to the time window and duration constraint and move operators in amortized constant time. For large problems, geometric and structural problem decompositions are used. Comparing the existing methods on classical benchmark VRP and its variants, their proposed algorithm outperforms and it is simple and efficient.

Baños *et al.* [45] propose a multi objective procedure based on Simulated Annealing algorithm to tackle a multi objective variant of the VRPTW. The objective is to minimize the travel distance and the imbalance of the routes with respect to distance travelled by vehicle and the loads delivered by them.

The Vehicle Routing Problem with Time Deadline (VRPTD) is a VRPTW where earliest arrival time is relaxed. Customers are supposed to be served before due times. Thangiah and Osman [46] develop the Deadline Sweep Heuristic, the Push-Forward Insertion Heuristic and the Genetic Sectoring Heuristic. A local post optimization is used

to improve solution that solved by these method. Park [47] use bi-criteria-saving algorithm (BC-saving algorithm) for solving VRP with time and area-dependent travel speeds. He aims to minimize total vehicle operation time and total weighted tardiness. In his following study [48], a hybrid genetic algorithm with a greedy interchange local optimization algorithm is presented. This algorithm provide better solutions then BC-saving algorithm but it needs much longer CPU time. Kang *et al.* [49] propose a mixed integer programming formulation and a TS heuristic for VRPTD. The neighborhood is generated by using route perturbed and route improvement method. Karlaftis *et al.* [50] solve VRP with pick-ups, deliveries and time deadlines by using hybrid genetic algorithm.

3. PROBLEM DESCRIPTION

In this chapter, two mathematical programming formulations based on VRPTW and a single commodity flow constraints of VRPTD and the details of the problem are provided. We describe the assumptions of the problem according to the real operations of a security carrier company in Turkey. The related parameters and decision variables are also introduced.

3.1. Problem Formulation

In this problem, we consider a security carrier company, which provides cash transportation between customers or demand points within Istanbul to minimize transportation cost. The company has its own fleet of vehicles. The amount of money is carried by a vehicle along a route from the depot to a set of geographically dispersed customers with known demand. During the day, each customer's requests must be satisfied before their last arrival time. The security carrier currently plans their vehicle routes manually every day based on personal experience and without considering a system perspective. This problem is characterized as a VRPTD.

VRPTD is a special VRPTW where earliest arrival time to the customers is relaxed. Each vehicle departs from the depot and satisfies requests of successive customers and returns to the same depot. The maximum tour duration restricts each vehicle to complete its assignment in a limited time.

In our problem, each customer has a known nonnegative demand and known nonnegative service time, and time deadline $[0, l_i]$ constraint where l_i is the latest arrival time. Each customer must be arrived before l_i by exactly one vehicle that must stay at that customer during the service. The violation of the time deadline constraint is not allowed.

On the other hand, we have the following assumptions according to the real operations of the security carrier:

- The daily amount of cash conveyance, and the number of customers served within certain time windows are known one day earlier of the operation.
- Currently they have a fixed number of vehicles but the security carrier determines its fleet size by intuition based on the number of customers served. However, the fleet size has to be examined from on optimization perspective. The cost of vehicles or drivers is not considered.
- The total fuel cost is considered as the transportation cost.
- The carriers provide a fixed service at the customers. Thus, the service time at the customers served is known and constant.
- Each customer is visited before its time deadline.
- Each customer is served once by exactly one vehicle within a tour.
- Each vehicle has to depart from a depot and return to the same depot by obeying capacity and maximum tour duration constraints.
- Distances between customers are asymmetric.
- Vehicle speed assumed to be fixed for all vehicles. Thus, travel durations between customers are the same regardless of vehicles.

The real travelling distances, the capacity of the vehicles, customers demand, time deadline, and service times and constitute input data for the problem. The maximum tour duration is considered as the time deadline of depot. The travelling times are calculated by dividing travelling distances by the speed of the vehicles.

The objective of our problem is to minimize the total transportation costs.

3.2. Mathematical Model of the Problem

We propose two mixed integer linear models for the VRPTD.

The first mathematical model is derived from the formulation of VRPTW. We relax the earliest arrival times of the customers. The following index sets, parameters and decision variables are used in the first model of the VRPDT.

Table 3.1. Index sets in the first model of VRPTD.

Sets	Definition
V	Set of vehicles
N	Set of demand points

Table 3.2. Indices in the first model of VRPTD.

Indices	Definition
i, j	Index for customers and depot ($i, j \in N \cup \{0\}$)
k	Index for vehicles

Table 3.3. Parameters in the first model of VRPTD.

Parameters	Definition
a_i	Demand of customer i
c	Traveling cost per unit distance
d_{ij}	Asymmetric distance from customer i to j
l_i	Latest arrival time at customer i
s_i	Service time for customer i
t_{ij}	Asymmetric travelling time between customers i and j
q	Capacity of each vehicle
M	A large number

Table 3.4. Variables in the first model of VRPTD.

Variables	Definition
R_{ik}	Arrival time at customer i
X_{ijk}	Binary variable whether arc (i, j) traversed by vehicle k

In the mathematical model, it is assumed that the depot does not have demand. In addition, the service time of depot is zero for all vehicles. We assume that all vehicles depart from the depot at time zero. The maximum tour duration of each vehicle is considered as the latest arrival time of the depot.

$$c\sum_{i\in I}\sum_{j\in I} d_{ij}X_{ij} \tag{3.1}$$

$$\sum_{k\in V}\sum_{j\in N\cup\{0\}} X_{ijk} \le 1, \forall i \in N \tag{3.2}$$

$$\sum_{i\in N} a_i \sum_{j\in N\cup\{0\}} X_{ijk} \le q, \forall k \in V \tag{3.3}$$

$$\sum_{j\in N} X_{0jk} \le 1, \forall k \in V \tag{3.4}$$

$$\sum_{i\in N} X_{i0k} \le 1, \forall k \in V \tag{3.5}$$

$$\sum_{i\in N\cup\{0\}} X_{ihk} - \sum_{j\in N\in\{0\}} X_{hjk} = 0, \forall h \in N, \forall k \in V \tag{3.6}$$

$$R_{ik} + s_i + t_{ij} - M(1-X_{ijk}) \le R_{jk}, \forall i \in N\cup\{0\}, \forall j \in N, \forall k \in V \tag{3.7}$$

$$R_{ik} + s_i + t_{ij} + M(1-X_{ijk}) \ge R_{jk}, \forall i \in N\cup\{0\}, \forall j \in N, \forall k \in V \tag{3.8}$$

$$R_{ik} + s_i + t_{i0} \le l_0, \forall i \in N, \forall k \in V \tag{3.9}$$

$$R_{ik} \le l_i, \forall i \in N, \forall k \in V \tag{3.10}$$

$$R_{ik} \ge 0, \forall i \in N\cup\{0\}, \forall k \in V \tag{3.11}$$

$$X_{ijk} \in \{0,1\}, \forall i, j \in N\cup\{0\}, \forall k \in V \tag{3.12}$$

Objective function (3.1) aims to minimize the total transportation cost function. Constraint (3.2) ensures that each demand point is visited exactly once. Constraint (3.3) shows that no vehicle can be loaded more than its capacity. Constraints (3.4) and (3.5)

enforce the vehicle start its route from depot and terminate its route at the depot. This also helps to minimize the number of vehicles used. With the constraint (3.6), these constraints guarantee each vehicle departs from the depot 0, after visiting the demand points it arrives at the depot, which means that no vehicle can stay at a demand point more than its service time. Thus, the constraint (3.6) indicates the flow conservation of the vehicle. The constraints (3.7) and (3.8) state the successive arrival times between demand points. If vehicle k departs from i and arrive at j, it cannot arrive at j before sum of the arrival time and service time of the demand point i, and travelling time (i, j) that is $R_{ik} + s_i + t_{ij}$. Here, *M* is a large scalar that ensures vehicle k is traveling from i to j. The total duration of each tour is limited by constraint (3.9) that is latest arrival time of depot. It makes sure that the maximum tour duration is satisfied for the each vehicle. Constraint (3.10) restricts the arrival times of customers to be prior time deadlines. In the point of the time windows, this constraint relaxes the earliest arrival time, and allows vehicle to perform immediately its service to a demand point even if it arrives too early. Non-negativity restriction of arrival time of each demand point and depot is indicated in constraint (3.11). Binary integrality is stated in constraint (3.12).

The second formulation is based on continuous flow variables and incorporate the Gavish-Glaves single commodity flow constraints [51]. The flow of the commodity along the arcs travelled by the vehicle is defined by continuous variables. Aras *et al.* [52] and Aksen *et al.* [53] are also used this variables in their models, as well. By using these variables and adding the time deadline constraints from the first model, we propose the second mathematical model for the VRPTD.

The following index sets, parameters, and decision variables are used in the second model of the VRPDT.

Table 3.5. Index sets in the second model of VRPTD.

Sets	Definition
I	Set of *n* customers and the depot denoted as 0, $I = 0, 1, \dots, n$
IC	Set of *n* customers only (a subset of I), $IC = 1, \dots, n$

Table 3.6. Indices in the second model of VRPTD.

Indices	Definition
i, j	Indices for customers and depot ($i, j \in I$)

Table 3.7. Parameters in the second model of VRPTD.

Parameters	Definition
a_i	Demand of customer i
c	Traveling cost per unit distance
d_{ij}	Asymmetric distance from customer i to j
l_i	Latest arrival time at customer i
s_i	Service time for customer i
t_{ij}	Asymmetric travelling time between customers i and j
q	Capacity of each vehicle
M	A large number

In the mathematical model, it is assumed that the depot does not have demand and the service time of depot is zero for all vehicles. We assume that all vehicles depart from the depot at time zero. The maximum tour duration of each vehicle is considered as the latest arrival time of the depot. That is to say, each vehicle must terminate its tour before the latest arrival time of the depot. That is:

$$R_0 = s_0 = 0$$

Table 3.8. Variables in the second model of VRPTD.

Variables	Definition
R_i	Arrival time at customer i
F_{ij}	Amount of flow from customer i to j
X_{ij}	Binary variable whether arc (i, j) traversed by a vehicle

$$\min c\sum_{i\in I}\sum_{j\in I} d_{ij}X_{ij} \tag{3.13}$$

$$\sum_{j\in I} F_{ij} - \sum_{j\in I} F_{ji} \le a_i,\ \forall i \in IC \tag{3.14}$$

$$F_{ij} \le (q - a_j)X_{ij}, \forall i, j \in I, i \ne j \tag{3.15}$$

$$F_{ij} \le q - a_j, \forall i \in I, \forall j \in IC, i \ne j \tag{3.16}$$

$$F_{ij} \ge a_i - M(1 - X_{ij}), \forall i \in IC, \forall j \in I, i \ne j \tag{3.17}$$

$$\sum_{j\in I,\, j\ne i} X_{ji} = 1, \forall i \in IC \tag{3.18}$$

$$\sum_{j\in I,\, j\ne i} X_{ij} = 1, \forall i \in IC \tag{3.19}$$

$$\sum_{i\in IC} X_{i0} = \sum_{i\in IC} X_{0i} \tag{3.20}$$

$$R_i + s_i + t_{ij} - M(1 - X_{ij}) \le R_j, \forall i \in I, \forall j \in IC \tag{3.21}$$

$$R_i + s_i + t_{ij} + M(1 - X_{ij}) \ge R_j, \forall i \in I, \forall j \in IC \tag{3.22}$$

$$R_i \le l_i, \forall i \in IC \tag{3.23}$$

$$R_i + s_i + t_{i0} \le l_0, \forall i \in IC \tag{3.24}$$

$$R_i \ge 0, \forall i \in I \tag{3.25}$$

$$F_{ij} \ge 0, \forall i, j \in I \tag{3.26}$$

$$X_{ij} \in \{0,1\}, \forall i, j \in I \tag{3.27}$$

Objective function (3.13) aims to minimize the total transportation cost function. Constraint (3.14) is the flow balance constraint for customers. Constraint (3.15) and (3.16) show that when arc (i,j) is traveled by a vehicle, vehicle capacity must be sufficient to satisfy the demand of customer j. With this respect, they provide upper bounds on the flow variables F_{ij}. Constraint (3.17) ensures that if arc (i,j) is traveled by a vehicle, all demand of customer i must be satisfied. Constraint (3.18) and (3.19) are outgoing and ingoing balance constraints for each customer i, ensuring that each customer is visited exactly once by only one vehicle. The depot's degree balance constraint (3.20) enforces that all vehicles departed from the depot must return the depot. Constraints (3.21) and (3.22) state that the successive arrival times between customers. If a vehicle departs from i and arrive at j, it cannot arrive at j before sum of the arrival time and service time of the customer i, and travelling time (i,j) that is $R_i + s_i + t_{ij}$. Here, M is a large scalar that ensures the vehicle travels from i to j (here, we assume that ($M = l_0$). Constraint (3.23) restricts the arrival times to be prior time deadlines. From the perspective of the time windows, this constraint relaxes the earliest arrival time, and allows a vehicle to perform immediately its service at a customer if it arrives too early. Constraint (3.24) makes sure that the maximum tour duration is satisfied for the each vehicle. Non-negativity restriction of arrival time of each customer and depot, the amount of flow from customer i to j is indicated in constraints (3.25) and (3.26). Binary restriction on decision variable X_{ij} is stated in constraint (3.27).

The following inequality enforces that the number of vehicles must be sufficient to carry all demand by considering the total vehicle capacity and all customers demand. This constraint is valid for our problem and it helps to reduce CPU times for CPLEX.

$$q\sum_{i\in IC} X_{0i} \geq \sum_{i\in IC} a_i \tag{3.28}$$

As can be seen easily, the problem is difficult to solve, even in polynomial time. The exact methods perform very poorly with respect to computational time even if the problem size is small and for large problems, it cannot find a feasible solution. Thus, an efficient heuristic has to be developed in order to handle the problem.

4. SOLUTION PROCEDURE AND EXPERIMENTAL DESIGN

Obtaining an optimal solution of the VRP in a reasonable amount of time is difficult since VRP is an NP-complete problem. VRPTD is more difficult than the VRP. Therefore, the exact methods can be used for the small instances but for the large instances, obtaining a feasible solution is not possible even with the high computational power. Therefore, for large problems, heuristics and metaheuristics have commonly been developed by researchers to find near-optimum solutions in a reasonable amount of time.

For the solution of the VRPTD with asymmetric distances, we implement a granular TS heuristic which includes CW Savings algorithm and Local Search (LS) algorithm embedded into it. Seven move operators are used to generate the solutions in the neighborhood of the current solution. The initial solution required by the TS heuristic is generated with CW method. For each feasible solution obtained during the progress of the algorithm, we apply a series of LS operators. In this section, we explain details of the TS algorithm designed in this study.

4.1. Tabu Search

TS is a local search metaheuristic algorithm that is one of the most effective heuristic for tackling the VRPTW. The algorithm starts with a feasible or infeasible solution. It generates neighborhood, $N(s)$, using a set of move operators for current solution and moves from the current solution to the best solution in the $\mathrm{N(s)}$ at each iteration. To avoid cycling, the attributes of the previous move is kept in a *tabu list* and the move remains tabu for a certain number of iterations called *tabu-tenure*. At the end of each iteration, the tabu tenure of moves is updated. The best neighborhood solution is chosen as the new current solution as long as it is not labeled as *tabu*, or although it is in the tabu list, it is better than the overall best feasible solution, i.e. *incumbent solution* obtained so far. This is called the *aspiration criterion*. The procedure is continued until termination criteria have been met.

4.1.1. Initial Solution

We make use of the savings heuristic of Clarke and Wright (1964) to generate an initial solution for the TS heuristic. This heuristic starts with the assumption that there is unlimited number of vehicles, so each customer is visited by a separate vehicle. The savings for all pairs of the customers i and j are calculated by Equation 4.1:

$$S_{ij} = t_{i0} + t_{0j} - t_{ij} \tag{4.1}$$

We also experiment parametric savings heuristic. We use Equation 4.2 proposed by Kuban and Öncan [55] for calculating the savings for all pairs of the customers i and j and compare results.

$$S_{ij} = t_{i0} + t_{0j} - \lambda t_{ij} + \varphi \,|\, t_{0i} - t_{j0} \,| + \eta \frac{a_i + a_j}{\bar{a}} \tag{4.2}$$

Where λ is route shape parameter, φ is the term used for considering asymmetry between customers with respect to their distances to the depot, and η is the parameter using for considering customer demands while calculating savings and lastly, $\bar{a}$ is the average demand used to normalize with the Equation 4.3:

$$\bar{a} = (1/n)\sum_{i \in IC} a_i \tag{4.3}$$

The pair which yields the highest savings is merged if:

- i and j are on the different tour
- neither i and j are interior to an existing tour

- the sum of demand in the tour covering *i* and *j* does not exceed the vehicle capacity
- the total tour duration covering *i* and *j* does not exceed the maximum tour duration
- the latest arrival time of the customers in the tour covering *i* and *j* is not violated

We repeat the algorithm until no more arcs satisfy these conditions. Since we work with asymmetric travelling distances, we have to consider reversed tours. Thus, the pair which satisfies the first three conditions is checked if the clockwise tour is better than the counterclockwise. If clockwise tour is better, and the last two conditions are satisfied, merging is carried out. If it is not better, the counterclockwise tour is checked. If counterclockwise tour is feasible, the reversed tour is accepted.

For each pair, we check four cases for merging in CW algorithm. These cases depend on positions of customer *i* and *j* in their tours.

Case 1*:* If both customers *i* and *j* are the first customers in their tours: In this case, one of the tours must be reversed to be merged. We select the tour that yields a better improvement when it is reversed if this is feasible.

Case 2*:* If both customer *i* and *j* are the last customers in their tours: In this case, one of the tours must be reversed in order to be merged. We select the tour that yields a better improvement when it is reversed if this is feasible.

Case 3*:* If customer *i* is the last customer and *j* is the first one in their tours: In this case, there is no need for a reversal operation. The new tour is constructed such that customer *j* comes after customer *i*.

Case 4*:* If customer *i* is the first customer and *j* is the last one in their tours: In this case, there is no need for a reversal operation. The new tour is constructed such that customer *i* comes after customer *j*.

In each case, we check if reversing the new constructed tour yields a better solution. By doing this, we actually check eight cases.

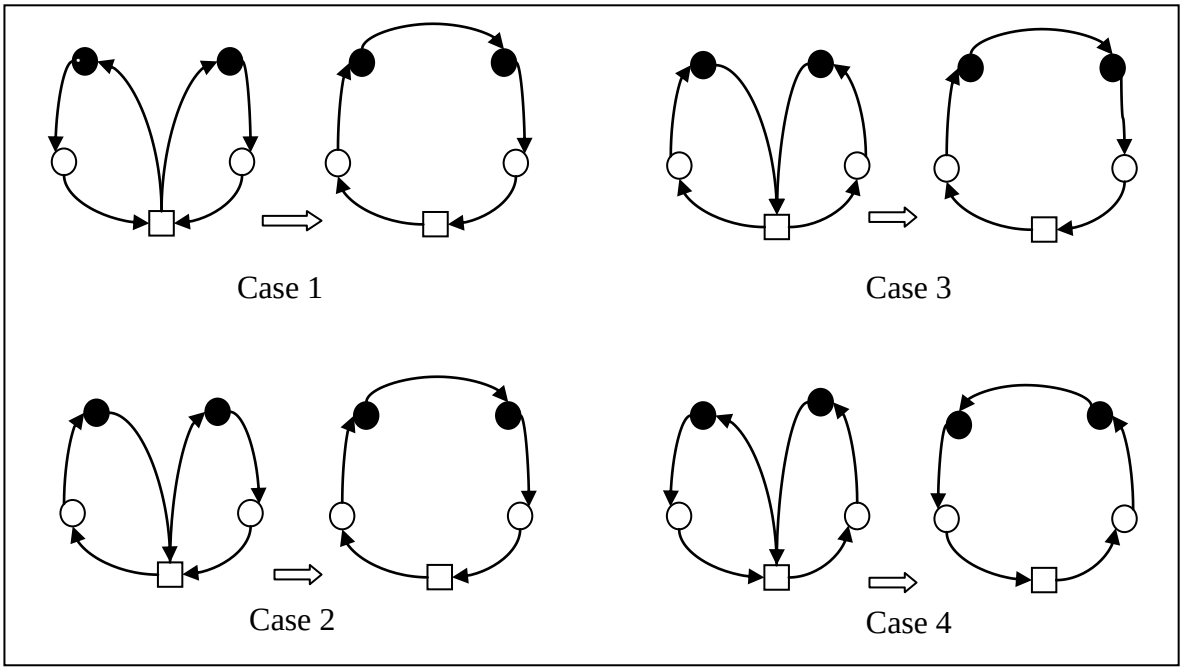

Figure 4.1. Illustration of merging cases for CW Savings algorithm.

For the initialization of TS algorithm, we use the Equation 4.1. We do not allow any infeasible merge. Thus, CW ends up with a feasible solution. A series of LS operators are applied to this solution for the intensification purposes. We will explain the LS operators later.

4.1.2. Granular Neighborhood Structure and Tabu Attributes

We implement granular neighborhood of Toth and Vigo [56] to our neighborhood structure. Granular neighborhood is used for reducing the computation time of neighborhood exploration. This does not affect the solution quality. The search of the neighborhood is restricted to insert *short arcs* only. To determine them to be evaluated in the neighborhood, the *granularity threshold* value proposed by Toth and Vigo [56] is calculated by Equation 4.4:

$$\nu = \mu \frac{z}{(n+K)} \tag{4.4}$$

where μ is a suitable positive sparsification parameter, and z is objective value of the initial solution, n is the number of customers and K is the number of vehicles obtained the initial solution. We choose the short arcs whose costs are not greater than this threshold value. By doing so, a sparse graph is determined.

At each iteration, the neighborhood of the current solution is generated by using some move operators. The move generator selects the arcs and the nodes from the generated sparse graph. In our problem, we use a multiple neighborhood based on seven move operators. These moves are 1-0 Move for the same tour and different tours (1-0st and 1-0dt), 2-0 Move for different tours (2-0dt), 1-1 Exchange Move for the same tour and different tours (1-1st and 1-1dt), 2-1 Exchange Move for the different tours (2-1dt), 2-2 Exchange Move for the different tours (2-2dt), 1-1-1 Rotation Move for different tours (1-1-1dt) and 2-Opt Move for the same tour and different tours (2-Opt-st and 2-Opt-dt). The working mechanism of the move operators can be described as follows:

1-0 Move: Select a customer and an edge from the same tour or from different tours. Remove the customer from its current position in its current tour and insert it between the customers that from the selected edge. If the edge and the customer are in the same tour and the customer is the only node in it, we select a different edge from a different tour. 1-0st does not cause any capacity change, but may result in the violation of maximum tour duration for the vehicle and latest arrival time constraints for the customer. 1-0dt may also result in the violation of the same constraints.

2-0 Move: Select successive two customers and an edge from the same tour or different tours. Remove the successive customers from their current position in their tour and insert them between the customers that constitute the selected edge. If the edge and the customers are in the same tour, we select a different edge from a different tour. This move operator may also violate the capacity and maximum tour duration constraint for the vehicles and latest arrival time constraints for the customers.

1-1 Exchange Move: Select two customers from different tours or from the same tour and swap their positions. If the selected customers are in different tours and the tours have only these customers, we do not allow swapping them since this does not generate a new

neighboring solution. Depending on the current tours that covering selected customers, this move may also result in exceeding the capacity and maximum tour duration of vehicles and the violation of customers' latest arrival time.

2-1 Exchange Move: Select successive two customers in any tour that has at least two customers, and a customer from a different tour that has also at least two customers. Swap the positions in the tours of these customers. This move may also violate the capacity and maximum tour duration constraints the vehicles and time deadline constraints of the customers.

2-2 Exchange Move: Select two successive customers in any tour that has more than one customer, and other two successive customers in a different tour that has more than two customers. Exchange the positions in the tours of these four customers. This move may also violate the capacity and maximum tour duration constraints of the vehicles and time deadline constraints of customers.

1-1-1 Rotation Move: Select three customers in different tours with at least two customers. Relocate the first customer to the position of second one, relocate the second one to the position of the third one, and relocate the third one to the position of first customer. This move may also violate the capacity and maximum tour duration constraints of the vehicles and time deadline constraints of the customers.

2-Opt Move: Select two arcs from the same tour with at least three customers or from different tours. Remove the arcs. If the arcs are in the same tour, connects the head node of the first arc to the tail node of the second one, and connect tail node of the first arc to head node of second arc. Selected arcs must be node-disjoint. If the arcs are in different tours, while holding the succeeding order of head customer of arcs same, we swap the following tour segments. In the different tour case, one of the tours must have at least two customers and the other one must have at least one customer. This move may violate the capacity and maximum tour duration constraints of the vehicles and time deadline constraints of the customers.

We illustrate these move operators in Figure 4.1 [57].

For the generation of neighboring solutions from the current solution, we apply all possible 1-0 moves and 1-1 Exchange moves. We implement other move operators in such a way that the probability of each move is inversely proportional to the number of times that move is used in the previous iteration. Then, with respect to these probabilities, a move operator is determined; and a number of solutions are generated. The best neighbor is selected as the new current solution if it is not tabu or although it is labeled as tabu, its cost is better than the incumbent solution. Then, we calculate the cost of the reversed tour(s). If it yields a better solution, we perform this solution as a new current solution. The number of neighbors generated to obtain next current solution is determined by a parameter. This parameter is set according to the problem size.

The probabilities of choosing the moves are updated at each iteration by Equation 4.5 proposed by Aksen *et al.* [58]:

$$prob_i = \frac{\sum_{j=1,\, j\neq i}^{5} num_j}{2(\sum_{j=1}^{5} num_j)} \tag{4.5}$$

where i is one of the five moves, $prob_i$ is probability of choosing move i, num_j is the total number of times move j is used so far in previous iterations. We update $prob_i$ at each iteration to balance the selection frequencies of moves.

We employ tabu restriction to avoid local minima and cycles. Thus, whenever we update the current solution according to a certain neighborhood structure, we create a tabu restriction for that structure. A move stays as tabu during the time they are tabu active. The tabu active status of each move is randomly determined in the interval [5,15] independent of the problem size. The tabu restrictions associated with all seven neighborhood structures is as follows:

1-0 Move: If a customer i is inserted between customer j and $(j+1)$, then the 1-0 move cannot relocate i to its previous position.

2-0 Move: If successive customers i and $(i+1)$ are inserted between customers j and $(i+1)$, then the 2-0 Move cannot relocate i and $i+1$ to their previous positions.

1-1 Exchange Move: If two customers i and j are swapped, then the 1-1 Exchange Move cannot be applied to them.

2-1 Exchange Move: If customers i, $(i+1)$ and j are swapped, the 2-1 Exchange Move cannot re-swap them.

2-2 Exchange Move: If customers i, $(i+1)$ and j and $(j+1)$ are swapped, the 2-2 Exchange Move cannot be applied to them.

1-1-1 Rotation Move: If customer i is located to customer j's position, and if j is located to customer k' s position, and k is located to i's position, then i, j, and k cannot be relocated to their previous positions by the 1-1-1 Exchange Move.

2-Opt Move: If 2-Opt Move is applied to customers i and j, the move cannot be applied again to the same customers.

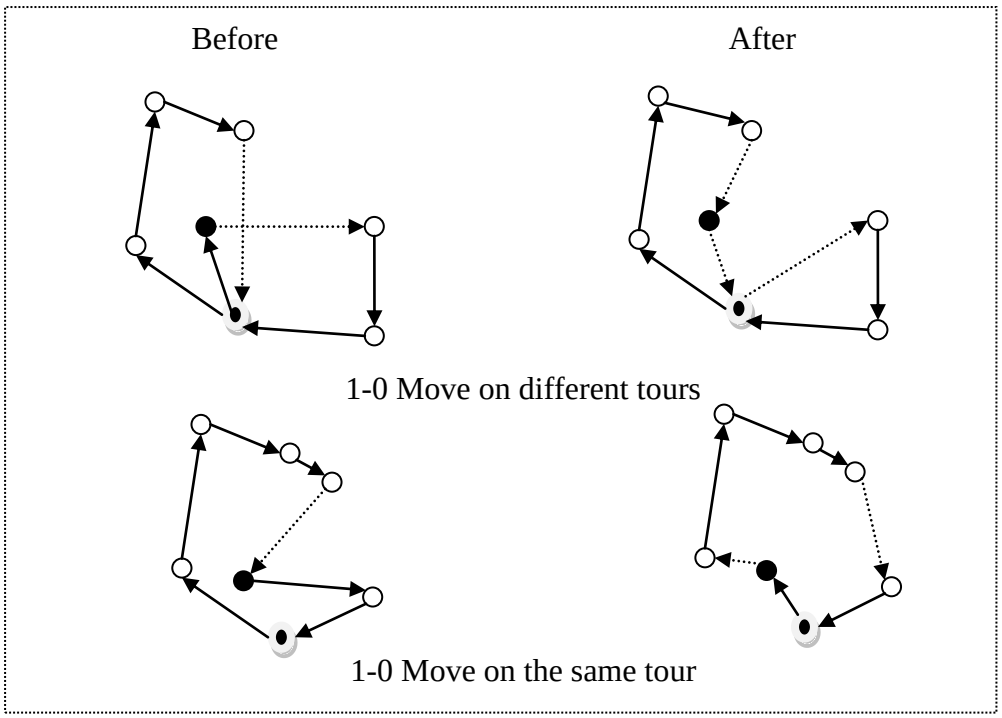

Figure 4.2. The move operators used to generate neighboring solutions.

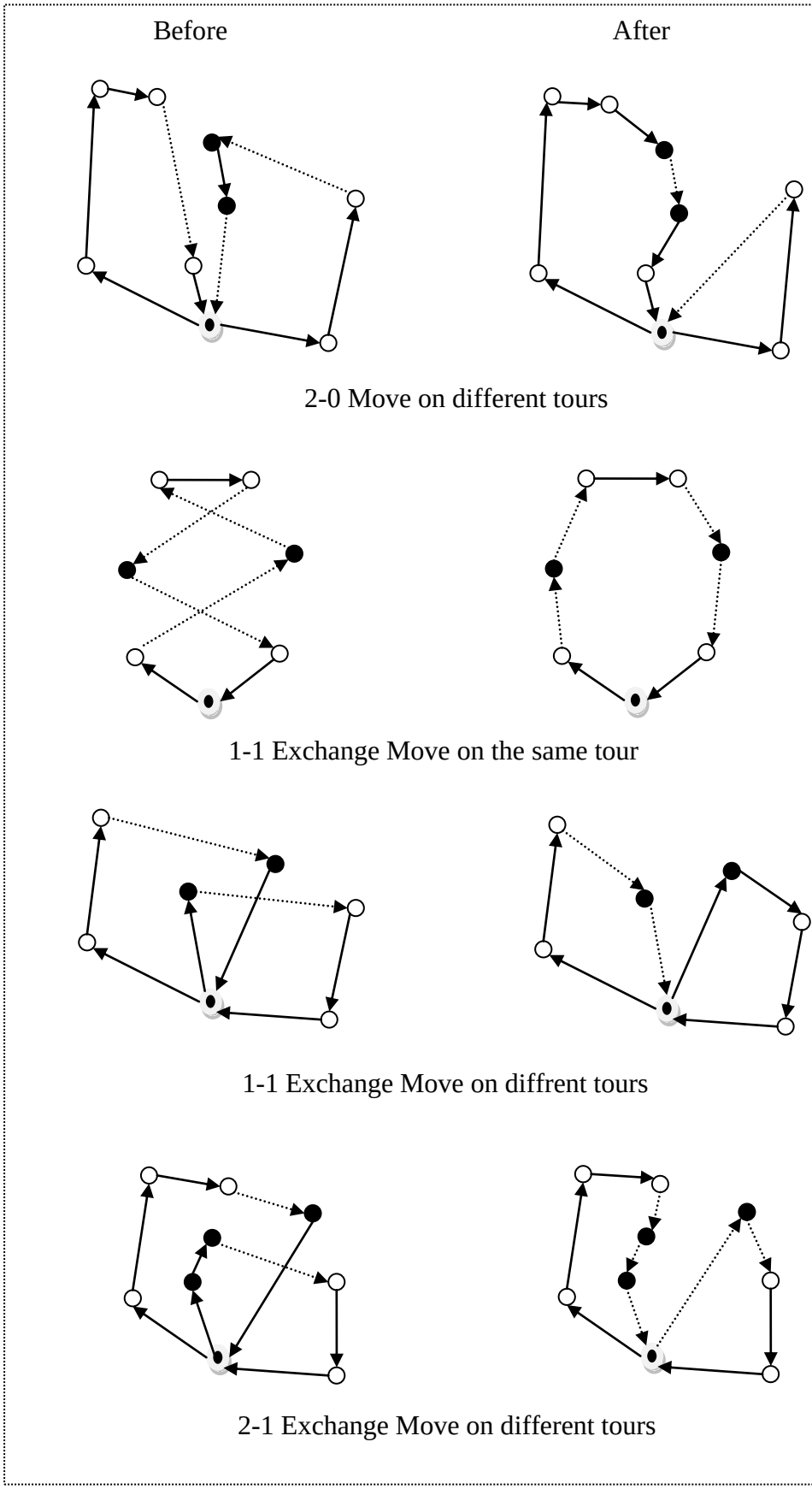

Figure 4.2. – (cont.).

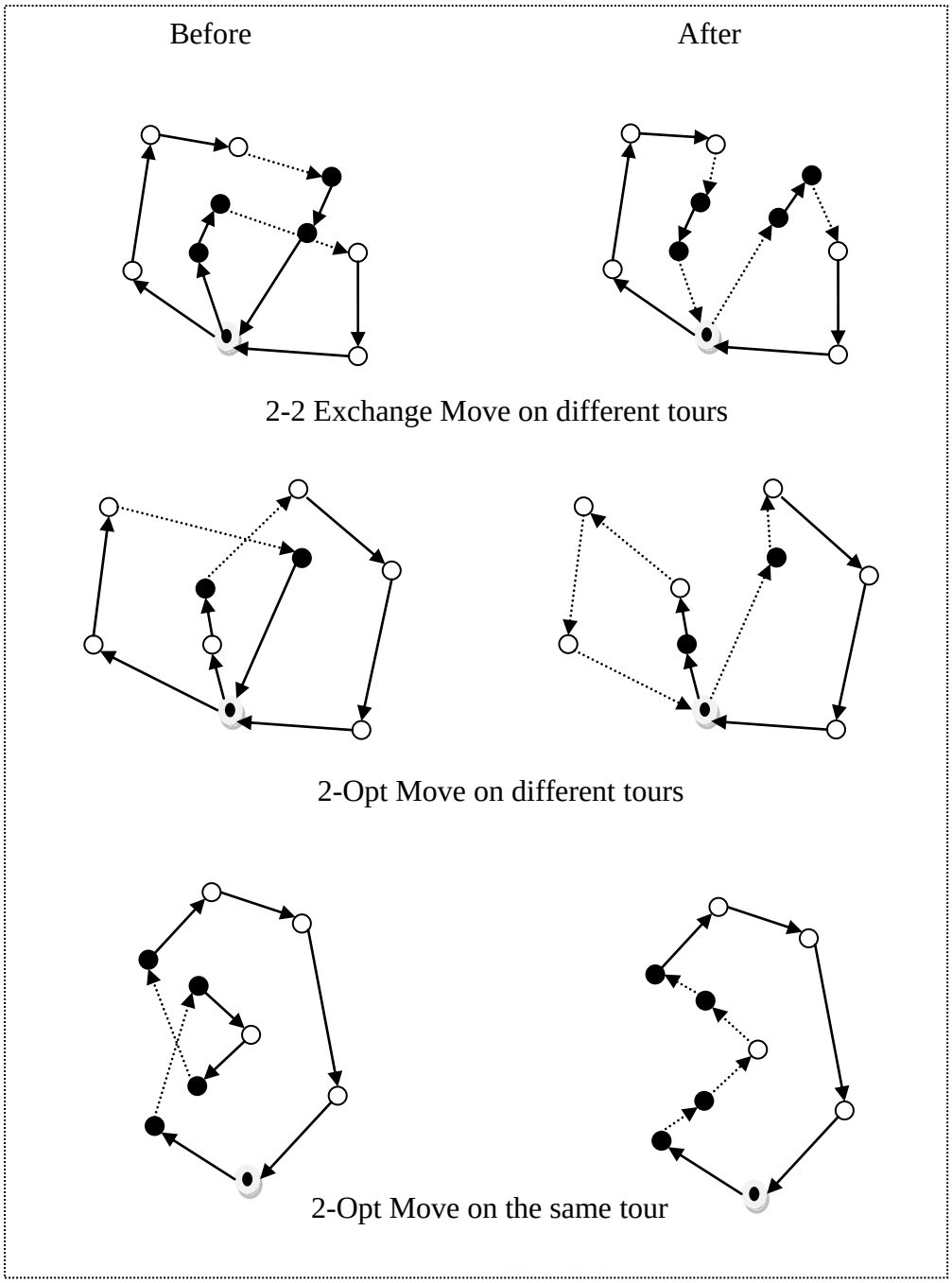

Figure 4.2. – (cont.).

As mentioned earlier, the initial solution of TS is feasible but neighborhood solutions are allowed to be infeasible with respect to the vehicle capacity, the maximum tour length, and the customer's latest arrival time. We penalize the infeasibility of the solutions, and in

the algorithm, a neighborhood solution is evaluated by means of the cost function that includes total transportation costs, weighted overload, and weighted tardiness, as follows:

$$z(s) = c(s) + \alpha\, Q(s) + \beta\, D(s) \tag{4.6}$$

where s denotes a solution, α, β are penalty factors of overload and overtime, respectively. The first component of the objective function is the total transportation cost. The second and third components are penalty terms. The first one is proportional to the excess total quantity of the vehicle capacity; the third one is proportional sum of the excess of the maximum tour duration and the latest arrival time of customers. If the solution is feasible, then the values of the objective function and the total transportation cost coincide.

At each iteration, the current solution is set to the neighborhood solution with the best cost function. We update the penalty factors of overload and overtime with Equation 4.7 and Equation 4.8 [59]:

$$\alpha = \begin{cases} \alpha(1+\delta) & \text{if the current solution is feasible wrt capasity} \\ \alpha/(1+\delta) & \text{if the current solution is infeasible wrt capacity} \end{cases} \tag{4.7}$$

$$\beta = \begin{cases} \beta(1+\delta) & \text{if the current solution is feasible wrt time deadline} \\ \beta/(1+\delta) & \text{if the current solution is infeasible wrt time deadline} \end{cases} \tag{4.8}$$

where δ is the parameter used to update α and β.

Moreover, for intensification of TS, a series of LS operators are applied to the initial solution and the current solution at the end of every 100 iterations, and also to the incumbent solution whenever it is updated [58].

4.1.3. Local Search

For the application of LS, we determine a sequence of move operators. The sequence is 2-Opt-st and 2-Opt-dt, 1-0st and 1-0dt, 2-0dt, 1-1st and 1-1dt, 2-1dt, 2-2dt, 1-1-1dt. Infeasible solutions are discarded during the LS process.

We experiment with the following three policies:

Best improvement of each node: We select a customer and search for the other customer that creates the highest improvement in the tour length with a LS operator. If we can find such a customer, associated LS operator is immediately performed. For each LS operator, we perform this procedure until all customers are examined.

The best improvement of all nodes: For each customer, we search for another customer that creates the highest improvement in the total distance with a LS operator. The associated LS operator is performed to the one that yields the highest improvement in the total tour length until determined total number iteration is satisfied. For each LS operator, we perform this procedure.

The best of all LS operators: For each customer, we search for another customer that creates the highest improvement in total tour length with all LS operators. The LS operator that yields the best improvement is performed. This procedure terminates when the number of maximum iterations is met.

Since the third case needs large computational times and does not provide better solution than the others do, we discard it. In LS algorithm, after applying the first case, we perform the second one in order to obtain more quality solutions.

4.1.4. Termination Criteria

The first termination criterion we use is the total number of iterations (max_{iter}) performed. The second criterion is the maximum number of iterations ($\max\_incumbent_{iter}$) during which the feasible incumbent or infeasible incumbent solution does not improve. We determine both values with respect to the problem size.

```
Determine initial routes such that each customer has served by a separate vehicle. For all pairs of
customers i and j, calculate the savings according to Equation 4.1.
While no more arcs satisfy the merging conditions
    Short the savings in non-increasing order
    Merge two trips starting with the best savings pair
    Check reverse of tour
    If the reversed tour is feasible and better than the current tour
    Reverse it
    End if
End while
Apply LS to the solution
Initialization of Tabu Search
    Set the current solution as the initial solution and the incumbent solution
    Determine granular neighbors
    While termination criteria are not met
        Update probability of each move operator according to Equation 4.3
        Find the probability interval for each move operator
        While neighborhood size is not satisfied
            Generate a number that is uniformly distributed
            Find the related move k with respect to this number
            Select a customer randomly among granular neighbors
            Generate a neighboring solution with considering customer by move k
            Calculate the infeasibility of the neighboring solution
            If new generated neighboring solution is not tabu or if it is tabu, but meets the
            aspiration criterion and it is also better than the previously generated one
                Update the best neighboring solution
```

Figure 4.2. Pseudocode of TS algorithm.

End if

End while

Calculate all neighboring solutions that can be generated by 1-0 Move and 1-1 Exchange Move

If new generated neighboring solution is not tabu or if it is tabu, but meets the aspiration criterion and it is also better than the previously generated one

Update the best neighboring solution

End if

Set the best neighboring solution as the current solution

Check the reverse of modified tours(s)

If the reversed of modified tour(s) makes an improvement on the current solution

Reversed the tour(s)

End if

Update tabu tenure of each tabu move

If the current solution is feasible and better than the incumbent solution

Apply LS to the current solution

Set the incumbent solution to the current solution

If the number of iteration is a multiples of 100

Apply LS to the current solution

End if

End while

Figure 4.2. – (cont.).

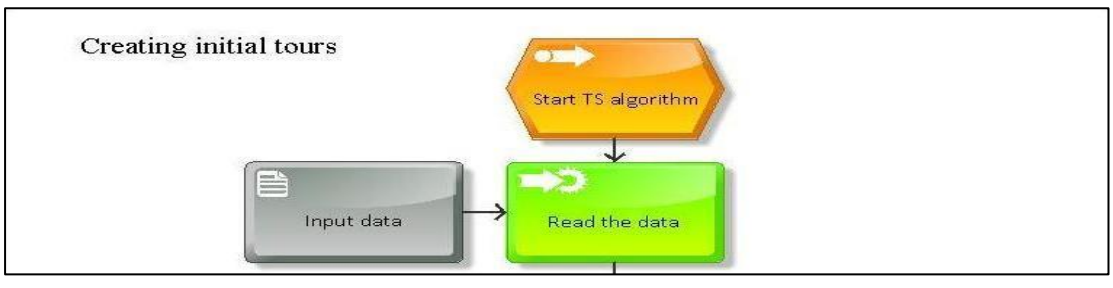

Figure 4.3. The flowchart of TS for solving the VRPTD with asymmetric distances.

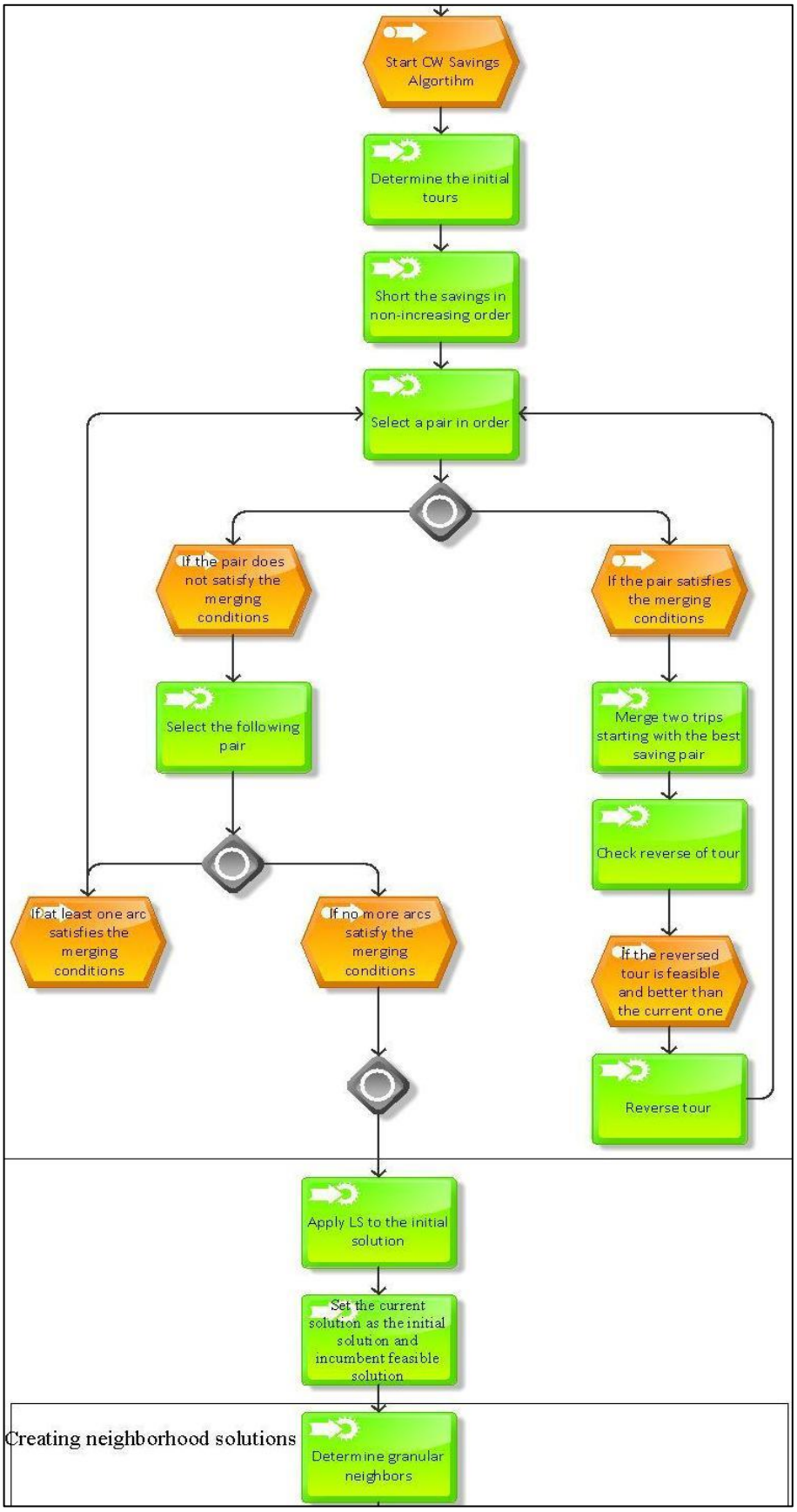

Figure 4.3. – (cont.).

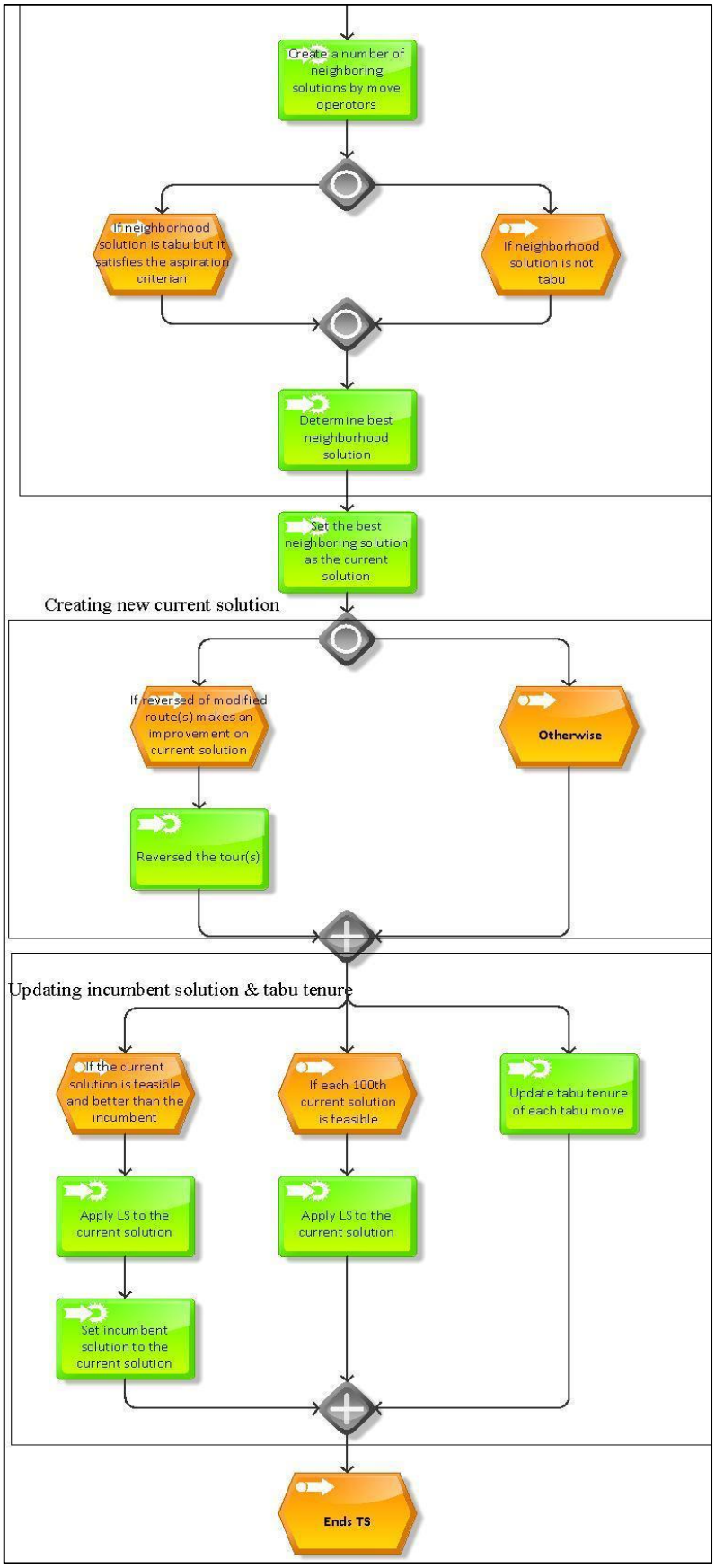

Figure 4.3. – (cont.).

5. RESULTS

5.1. Problem Generation

Since there are no benchmark instances for VRPTD with symmetric or asymmetric instances in the literature, we need to convert benchmark instances related to the other VRPs to our problem.

Seven of 14 classical VRP test problems due to Christofides *et al*. [60] (in short CMT) have maximum tour duration constraint. We can obtain a time deadline of each customer by using Equation 5.1 for converting the maximum tour duration to time deadline:

$$l_i = l_0 - (t_{i0} + s_i) \tag{5.1}$$

This equation says that the latest arrival time of each customer is equal to the difference between maximum tour duration, which is the latest arrival time of depot, and service time of that customer and travelling time from the customer to the depot. By doing this transformation, we can solve seven CMT problems, and compare our results with the best-known solutions in the literature. Besides, the distances in these seven data sets are Euclidean distances and therefore in order to provide consistent results we solve these problems as symmetric distances.

On the other hand, we randomly generate 42 asymmetric instances with the help of Solomon benchmark problems for VRPTW [61] For our problem we choose three of six Solomon benchmark problems which are called C2, RC1 and RC2. In the types of C2 customer locations are clustered while in RC types of problems, they are mix of clustered and random. C2 and RC2 type of problems have long horizon times and RC1 has short horizon times. Solomon test problems have time windows $[e_i, l_i]$, and the distances

between customers are Euclidean. However, we are only interested in the latest arrival times of customers, and asymmetric distances. To this end, we assume that ready times of all customers are zero. To convert symmetric distances to asymmetric one, we use Equation 5.2:

$$d_{ij}^{'} = \theta d_{ij} \tag{5.2}$$

where d'_{ij} is the new distance from customer i to j, and θ is a random number in the interval [1.2,1.4] and d_{ij} is the Euclidean distance. The distance values from i to j and from j to i can be different depending on the θ value.

5.2. Computational Environment

The proposed TS algorithm was coded in Microsoft Visual Studio 2010 C# language. The heurisric runs are performed on a server equipped with Intel® Xeon® CPU X5460 @ 3.16 GHz and 27.9 GB of RAM. In order to assess the quality of some solutions, we have solved the developed mathematical models using CPLEX 12.5. The CPLEX runs are executed on a server equipped with Intel Xeon X5460 3.16GHz Quad-Core predecessor and 16 GB RAM. The accuracy is controlled by a number of options in GAMS models. A relative optimality gap (OPTCR) is set to 0.00001. We also give there hours of resource usage. The server terminates to search for improving when the gap reaches the OPTCR's value or when time is exceed.

5.3. TS Algorithm Parameters

There are some parameters of the TS algorithm, which may have an effect on the solutions and CPU times. Their values are determined based on some preliminary experiments.

The most important attribute of the TS algorithm, the tabu tenure, is set to random integer values in the interval [5,15]. The maximum number of iterations is set to 5000 and the maximum number of iteration during which feasible or infeasible incumbent solution does not improve is set to 500.

The number of neighboring solutions to generate the next current solution varies with respect to the problem size. An increase in the size of neighboring solutions leads to larges solution space explored, but also results in higher CPU times. As mentioned earlier, for the generation of each neighborhood solution, we use seven move operators. We test these move operators on different problems and realize that the neighboring solutions generated by 1-0 Move or 1-1 Exchange Move mostly provide higher quality solutions compared to other moves. For this reason, we generate all possible neighbors by 1-0 Move and 1-1 Exchange Move. For the other moves, we only generate a subset of all neighboring solutions using the probability in Equation 4.5.

In order to determine short arcs to be used in the granular neighborhood space, we use Equation 4.4. As mentioned earlier, the granular neighborhood does not have an influence on the solution quality but it reduces the CPU times. The sparsification parameter μ is selected in the interval [0.5,5.0]. For Euclidean distances, we set μ to 1.5 and, for distances it is 3.5 since we consider an asymmetric distance matrix which means that in asymmetric distances two times more arcs are evaluated.

For penalty parameters, we receive guidance from the paper by Cordeau *et al.* [59]. We set $\alpha = \beta = 1$ as initial values and they are modified by Equations 4.7 and 4.8 at each iteration. The parameter δ used to update α and β does not have any effect on solution. Thus, we set it around 0.5.

For parameters using for calculating of savings by Equation 4.2, we receive guidance from the paper by Altınel and Öncal [55]. Values of parameters λ, φ, η used for CMT problems are listed in Table 5.1. For each random generated asymmetric problem, we experiment all these seven λ, φ, η values in Table 5.1 and take the values that give us the best solution.

Table 5.1. Parameters used for calculating savings.

Problem	λ	φ	η
C6	1.4	0.9	0.3
C7	1.2	0.2	0.8
C8	1.4	0.3	0.3
C9	1.6	0.3	0.4
C10	1.3	0.0	1.1
C13	1.1	0.1	0.3
C14	1.3	0.3	0.6

We set each vehicle speed to 60 km/hr.

Lastly, for the calculation of travelling cost, we use the current average mean fuel cost per kilometer of a security carrier vehicle.

5.4. Computational Results

In Table 5.2, the number of customers, the service time of customers, the capacity of the vehicles and maximum tour duration for CMT problems used in our thesis are illustrated.

Table 5.2. Data sets of CMT problem used for CVRPTD.

Problem	Number of Customer	Vehicle Capacity	Service Time	Maximum Tour Duration
C6	50	160	10	200
C7	75	140	10	160
C8	100	200	10	230
C9	150	200	10	200
C10	199	200	10	200
C13	120	200	50	720
C14	100	200	90	1040

Table 5.3. Comparison of the best known results of the CMT problems with CW, CW+LS and TS algorithms.

Problem	Best known	CW	CW+LS	Tabu	Percent Gap of CW (%)	Percent Gap of CW+LS (%)	Percent Gap of TS (%)
C6	555.43	618.38	618.38	589.28	11.33	11.33	6.09
C7	909.68	975.46	967.52	946.48	7.23	6.36	4.05
C8	866.75	973.94	961.39	908.33	12.37	10.92	4.80
C9	1164.52	1287.64	1268.58	1220.08	10.57	8.94	4.77
C10	1417.85	1538.66	1505.33	1505.33	8.52	6.17	6.17
C13	1545.98	1592.26	1572.59	1572.59	2.99	1.72	1.72
C14	866.37	875.75	867.17	867.17	1.08	0.09	0.09
			Average		7.73	6.50	3.96

Table 5.3 reveals the comparison of the best known results of the CMT problem with the CW, CW with LS and TS algorithm initialized by CW with LS. Here, a gap is defined as Gap = 100 x (Result of Algorithm-Best Known Result)/Best Known Result. We observe that LS algorithm improves the initial solution. The result obtained by CW+LS for problem C14 is close to the best-known solution. TS algorithm does not improve the initial solution for three of the seven problems. However, on average, the solutions obtained by TS algorithm turn out a better result than CW+LS algorithm. The average gap of our algorithm's final results is 3.96 %.

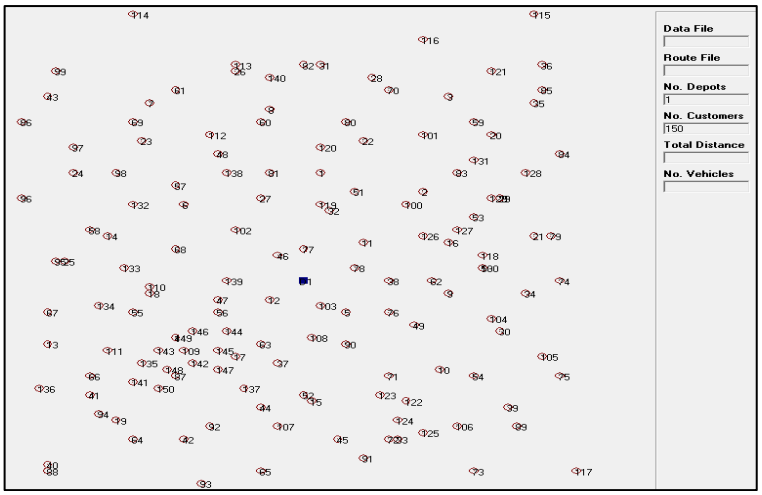

Figure 5.1. Visualization of customers' locations of CMT C9 problem.

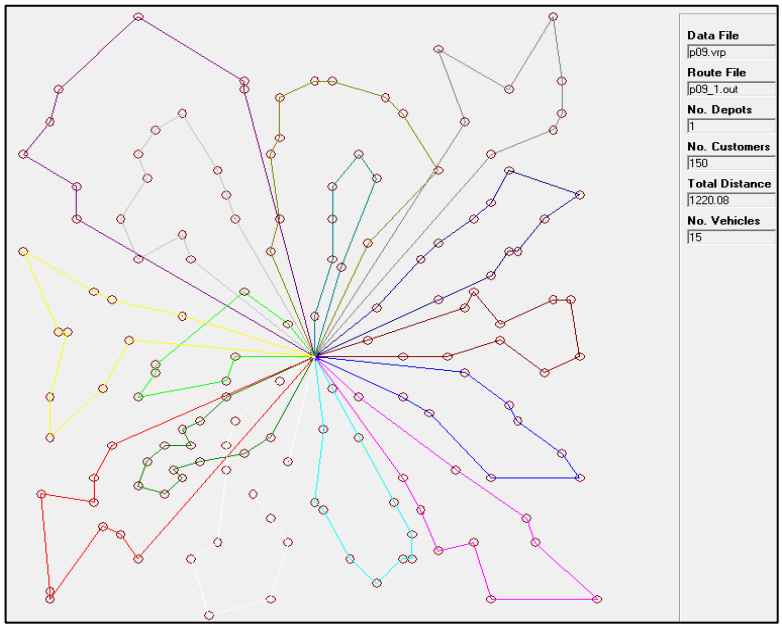

Figure 5.2. Visualization of tours of CMT C9 problem.

The visualization of customers' locations and the tours obtain by the TS algorithm for the CMT C9 problem are shown in Figure 5.1 and 5.2, respectively.

Table 5.4 shows the solutions of CMT problems solved by using parametric savings heuristics. To compare the solutions of classic and parametric savings heuristics, we also indicate the percentage gaps of each algorithm. The symbol '*' denotes the results of parametric savings heuristics.

Table 5.4. Solutions for CMT problems by using two type of savings' equations.

Problem	Best known	CW*	CW+LS *	Per. gap of CW* (%)	Per. gap of CW+LS* (%)	Per. gap of CW (%)	Per. gap of CW+LS (%)
C6	555.43	608.35	585.72	9.53	5.45	11.33	11.33
C7	909.68	965.50	964.07	6.14	5.98	7.23	6.36
C8	866.75	969.55	938.13	11.86	8.24	12.37	10.92
C9	1164.52	1272.45	1262.96	9.27	8.45	10.57	8.94
C10	1417.85	1500.65	1485.44	5.84	4.77	8.52	6.17
C13	1545.98	1591.02	1572.07	2.91	1.69	2.99	1.72
C14	866.37	874.87	869.96	0.98	0.41	1.08	0.09
		Average		6.65	5.00	7.73	6.50

For all CMT problems parametric savings heuristic yields better solutions than the classic savings heuristic.

The service time of customers, the capacity of the vehicles and maximum tour duration of asymmetric instances generated by Solomon benchmark problem are shown in Table 5.5:

Table 5.5. Data sets of problem.

Type of Instance	Service Time	Vehicle Capacity	Maximum Tour Duration
C2	90	500	3390
RC1	10	200	240
RC2	10	500	960

We take capacity of RC2 problems as 500 instead of 1000 since capacity has an effect on small problems.

In order to assess the efficiency of the developed mathematical models, we have solved the generated asymmetric problems by both of them using CPLEX 12.5. The results are listed in Table 5.6:

Table 5.6. Comparison of two mathematical models' solutions.

Problem	Number of Customers	Model I		Model II	
		CPLEX	Relative Gap	CPLEX	Relative Gap
C203	5	370.99	-	370.99	-
	10	716.43	-	716.43	-
	15	908.61	0.01	902.40	-
	20	987.31	0.14	986.77	-
	25	1129.02	0.18	1115.19	-
	30	1245.02	0.25	1243.99	-
	50	2613.42	0.46	1946.26	0.02
C204	5	370.99	-	370.99	-
	10	696.02	-	696.02	-
	15	902.40	0.01	902.40	-
	20	963.25	0.01	955.93	-
	25	1037.19	-	1037.19	-

Table 5.6. – (cont.).

Problem	Number of Customers	Model I		Model II	
		CPLEX	Relative Gap	CPLEX	Relative Gap
C204	30	1231.90	-	1231.90	-
	50	2105.02	0.33	1876.91	-
RC101	5	468.15	-	468.15	-
	10	967.02	-	967.02	-
	15	1091.01	-	1091.01	-
	20	1935.08	-	1935.08	-
	25	-	-	2122.29	-
	30	-	-	3319.58	-
	50	-	-	5415.58	0.25
RC108	5	429.95	-	429.95	-
	10	870.89	-	870.89	-
	15	975.88	-	975.88	-
	20	1508.45	-	1508.45	-
	25	1582.32	0.61	1579.24	-
	30	2240.32	0.65	2228.07	-
	50	-	-	3304.67	0.14
RC202	5	429.94	-	429.94	-
	10	725.00	-	725.00	-
	15	815.87	-	815.87	-
	20	1301.41	0.3	1301.41	0.01
	25	1362.10	-	1362.10	-
	30	1816.18	0.6	1816.18	-
	50	2765.59	0.64	2503.91	0.11
RC206	5	429.94	-	429.94	-
	10	725.00	-	725.00	-
	15	798.90	-	798.90	-
	20	1150.34	-	1150.34	-
	25	1397.71	0.58	1362.10	-
	30	1832.36	0.63	1816.18	-
	50	-	-	2318.19	3

We observe that the developed mathematical model II is more efficient than the first one. It can find optimal solutions for most problems, and it obtains a feasible solution for problems which cannot be solved by mathematical model I in a given time. We can obtain optimal values for 36 of 42 generated asymmetric instances by solving developed mathematical model II using CPLEX 12.5.

We also solve our random generated asymmetric instances derived from Solomon benchmark problems by the TS algorithm. The results of CPLEX, CW, CW with LS and TS over CW+LS are shown in Table 5.7:

Table 5.7. Solutions for generated asymmetric problem.

Problem	Number of Customers	CPLEX	CW	CW+LS	Tabu
C203	5	370.99	370.99	370.99	370.99
	10	716.43	716.43	716.43	716.43
	15	902.40	966.77	941.12	902.40
	20	986.77	1067.43	1020.48	986.77
	25	1115.19	1184.35	1129.03	1115.19
	30	1243.99	1443.43	1257.69	1257.69
	50	1946.26	2302.77	2089.40	2054.71
C204	5	370.99	370.99	370.99	370.99
	10	696.02	696.02	696.02	696.02
	15	902.40	931.22	913.15	902.40
	20	955.93	1006.17	969.02	955.93
	25	1037.19	1119.83	1037.19	1037.19
	30	1231.90	1286.22	1231.90	1231.90
	50	1876.91	2129.08	2006.70	1995.46
RC101	5	468.15	468.15	468.15	468.15
	10	967.02	1112.72	1081.03	967.02
	15	1091.01	1674.60	1657.27	1091.01
	20	1935.08	2213.97	2182.27	1935.08
	25	2122.29	2344.39	2250.33	2250.33
	30	3319.58	3980.65	3701.84	3701.84
	50	5415.58	6430.72	5726.05	5726.05
RC108	5	429.95	429.95	429.95	429.95
	10	870.89	895.97	870.89	870.89
	15	975.88	1008.19	977.31	975.88
	20	1508.45	1802.50	1785.52	1537.55
	25	1579.24	1906.15	1884.79	1579.24
	30	2228.07	2610.09	2536.31	2280.46
	50	3304.67	4062.84	3997.11	3820.34
RC202	5	429.94	429.94	429.94	429.94

Table 5.7. - (cont.).

Problem	Number of Customers	CPLEX	CW	CW+LS	Tabu
RC202	10	725.00	781.10	725.00	725.00
	15	815.87	822.17	815.87	815.87
	20	1301.41	1351.09	1307.47	1307.47
	25	1362.10	1429.97	1396.88	1396.89
	30	1816.18	1882.21	1822.86	1822.86
	50	2503.91	2593.98	2503.91	2503.91
RC206	5	429.94	429.94	429.94	429.94
	10	725.00	781.10	725.00	725.00
	15	798.90	798.90	798.90	798.90
	20	1150.34	1234.88	1150.34	1150.34
	25	1362.10	1436.69	1396.88	1396.88
	30	1816.18	1864.07	1832.37	1816.18
	50	2318.19	2443.82	2381.64	2357.80

We benchmark CW, CW+LS and TS algorithms with the objective value of the CPLEX solutions. In Table 5.8, gaps between CPLEX solutions and proposed algorithms are provided:

Table 5.8. Percentage gaps between the CPLEX solutions and CW, CW+LS and TS algorithms for generated asymmetric instances.

Problem	Number of Customers	Percent Gap of CW (%)	Percent Gap of CW + LS (%)	Percent Gap of TS (%)
C203	5	-	-	-
	10	-	-	-
	15	7.13	4.29	-
	20	8.17	3.42	-
	25	6.20	1.24	-
	30	16.03	1.10	1.10
	50	18.32	7.36	5.57
C204	5	-	-	-
	10	-	-	-
	15	3.19	1.19	-
	20	5.26	1.37	-
	25	7.97	-	-

Table 5.8. – (cont.).

Problem	Number of Customers	Percent Gap of CW (%)	Percent Gap of CW + LS (%)	Percent Gap of TS (%)
	30	4.41	-	-
	50	13.44	6.91	6.32
RC101	5	-	-	-
	10	15.07	11.79	-
	15	53.49	51.90	-
	20	14.41	12.77	-
	25	10.47	6.03	6.03
	30	19.91	11.52	11.52
	50	18.74	5.73	5.73
RC108	5	-	-	-
	10	2.88	-	-
	15	3.31	0.15	-
	20	19.49	18.37	1.93
	25	20.70	19.35	-
	30	17.15	13.83	2.35
	50	22.94	20.95	15.60
RC202	5	-	-	-
	10	7.74	-	-
	15	0.77	-	-
	20	3.82	0.47	0.47
	25	4.98	2.55	2.55
	30	3.64	0.37	0.37
	50	3.60	-	-
RC206	5	-	-	-
	10	7.74	-	-
	15	-	-	-
	20	7.35	-	-
	25	5.48	2.55	2.55
	30	2.64	0.89	-
	50	5.42	2.74	1.71
Average		8.62	4.97	1.40

In Table 5.8, a gap is defined as Gap = 100 x (Result of Algorithm- CPLEX solution)/CPLEX solution Result.

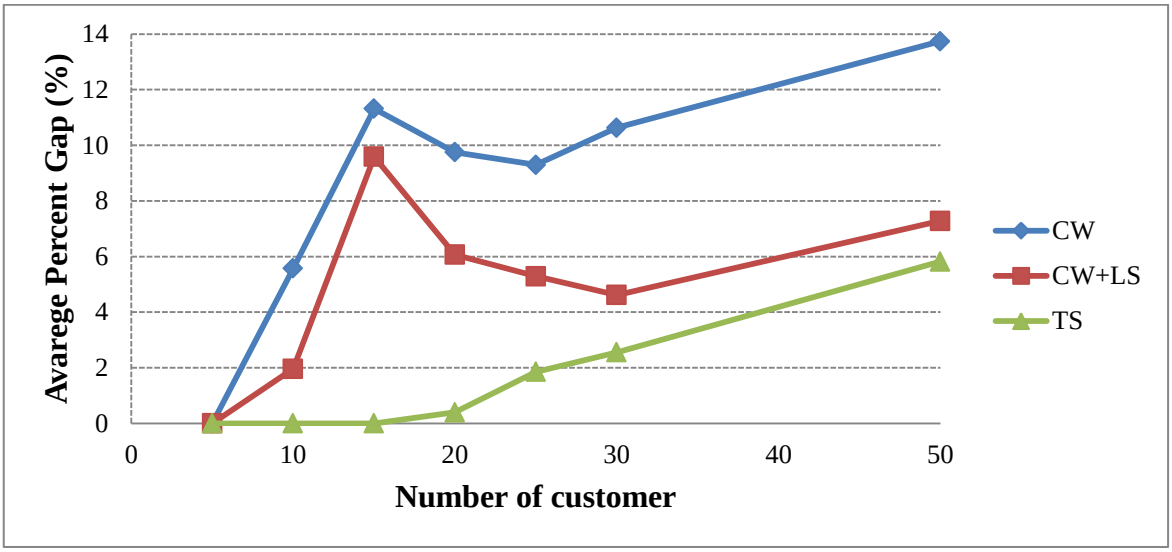

Figure 5.3. Percent gaps of algorithms over number of customers for 42 test instances.

When the solution methods are compared, for instances with five customers, CW algorithm can easily obtain the optimal solutions as CPLEX results. CW+LS algorithm yields more savings on objective value of CW solutions. The average saving is observed as 3.65%. The TS algorithm creates significant improvements on the initial solution for some problems. However, there are instances which it is not work well, too. The visualization of tours for RC202 with 50 customers is given in Figure 5.4:

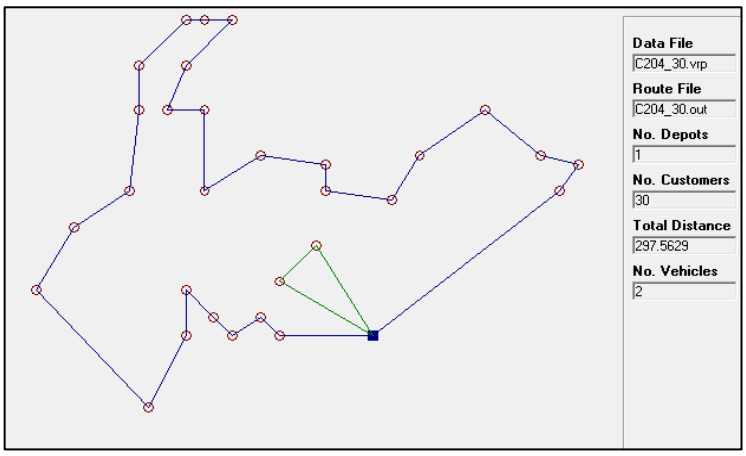

Figure 5.4. Visualization of tours for the problem C204-50.

Table 5.9 shows the solutions of random generated asymmetric problems solved by using parametric savings heuristics. To compare the solutions of classic and parametric

savings heuristics, we also indicate the percentage gaps of each algorithm. The symbol '*' denotes the results of parametric savings heuristics.

Table 5.9. Solutions for generated asymmetric problems by using two type of savings' equations.

Problem	Number of Customers	CW*	CW+LS*	Percent Gap of CW* (%)	Percent Gap of CW+LS* (%)	Percent Gap of CW (%)	Percent Gap of CW + LS (%)
C203	5	89.61	89.61	-	-	-	-
	10	173.05	173.05	-	-	-	-
	15	233.51	233.51	7.13	7.13	7.13	4.29
	20	248.03	246.49	4.06	3.42	8.17	3.42
	25	282.95	272.71	5.04	1.24	6.20	1.24
	30	303.60	301.27	1.04	0.26	16.03	1.10
	50	514.33	502.64	9.41	6.92	18.32	7.36
C204	5	89.61	89.61	-	-	-	-
	10	168.12	168.12	-	-	-	-
	15	221.82	220.57	1.77	1.19	3.19	1.19
	20	235.60	234.06	2.04	1.37	5.26	1.37
	25	256.25	250.73	2.28	0.08	7.97	-
	30	301.75	297.56	1.41	-	4.41	-
	50	492.35	475.97	8.60	4.99	13.44	6.91
RC101	5	113.08	113.08	-	-	-	-
	10	302.20	298.71	29.38	27.89	15.07	11.79
	15	398.62	394.58	51.26	49.73	53.49	51.90
	20	533.01	527.12	14.04	12.77	14.41	12.77
	25	598.86	598.40	16.82	16.73	10.47	6.03
	30	955.84	895.67	19.21	11.70	19.91	11.52
	50	1535.12	1434.69	17.35	9.68	18.74	5.73
RC108	5	103.85	103.85	-	-	-	-
	10	210.36	210.36	-	-	2.88	-
	15	241.21	235.72	2.33	-	3.31	0.15
	20	431.28	431.28	18.37	18.37	19.49	18.37
	25	456.23	456.23	19.60	19.60	20.70	19.35
	30	624.91	608.12	16.11	13.00	17.15	13.83
	50	981.36	922.79	22.94	15.60	22.94	20.95
RC202	5	103.85	103.85	-	-	-	-
	10	182.61	175.12	4.28	-	7.74	-
	15	199.60	197.07	1.28	-	0.77	-
	20	322.83	315.69	2.70	0.43	3.82	0.47
	25	340.50	337.41	3.49	2.55	4.98	2.55

Table 5.9. – (cont.).

Problem	Number of Customers	CW*	CW+LS*	Percent Gap of CW* (%)	Percent Gap of CW+LS* (%)	Percent Gap of CW (%)	Percent Gap of CW + LS (%)
	30	454.64	440.31	3.64	0.37	3.64	0.37
	50	626.57	604.81	3.60	-	3.60	-
RC206	5	103.85	103.85	-	-	-	-
	10	182.61	175.12	4.28	-	7.74	-
	15	196.03	192.97	1.59	-	-	-
	20	290.99	277.86	4.73	-	7.35	-
	25	347.03	337.41	5.48	2.55	5.48	2.55
	30	450.26	442.60	2.64	0.89	2.64	0.89
	50	574.72	559.95	2.64	-	5.42	2.74
		Average		7.39	5.44	8.62	4.97

For most problems, the solutions obtained by using parametric savings heuristic are better than classic savings algorithm. However, since there are large gaps between CPLEX results and CW+LS* results for RC101 problem sets, on average classical heuristic provides better final results than the parametric savings heuristic.

Since solution quality rather than computation time is critic issue for us, we do not aim to benchmark computational time for this study. Therefore, we give the computational times of proposed algorithm and CPLEX solutions for these problem instances on Appendix A.

5.5. Real Life Experiment

The TS algorithm is also applied to a real life data set of a security carrier company which we mention in section 3.1.

The company provides us with information about the locations of their depot and customers the capacity of a vehicle, demand and required service time of each customer, and the latest arrival time each customer. Since we are not provided with asymmetric distances we solve this problem as symmetric by using Euclidean metric.

Each day, vehicles can serve a different set of customers. A customer can be served more than one during the week but it must be served by only one vehicle during a day. The operation times start at 6:00 am and 09:00 am, and end at 11:30 am. Thus, the maximum tour durations are considered as [0,330].

Table 5.10. Data sets of the security carrier company's problems.

Weekdays	Number of Customers	Vehicle Capacity
Monday	86	80
Tuesday	77	80
Wednesday	95	80
Thursday	95	80
Friday	98	80

The locations of all customers in Istanbul are illustrated in Figure 5.5:

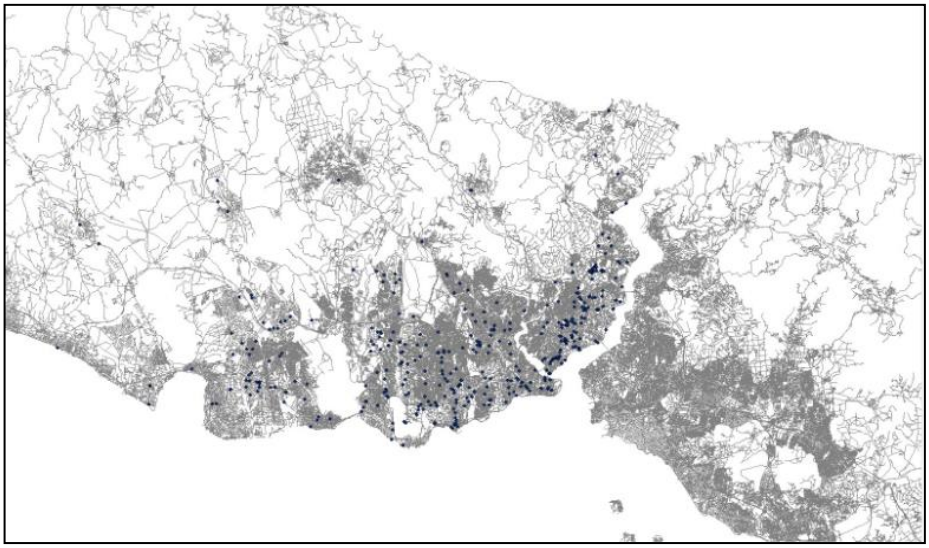

Figure 5.5. Locations of all customers of the security carrier company.

The solutions of the problem sets are shown in Table 5.11 and 5.12:

Table 5.11. Solutions based on weekdays for the security carrier company's problem.

Weekdays	Total cost	Number of vehicle
Monday	1625.37	5
Tuesday	724.16	4

Table 5.11. – (cont.).

Weekdays	Total cost	Number of vehicle
Wednesday	1650.29	5
Thursday	1879.00	5
Friday	1625.88	5

Table 5.12. Detailed solutions based on tours for the security carrier company's problem.

Weekdays	Vehicle ID	Number of customer served	Total demand of vehicle	Total cost
Monday	1	20	80	109.02
	2	20	80	131.00
	3	20	80	221.58
	4	6	80	39.44
	5	20	24	1113.66
Tuesday	1	20	80	132.75
	2	20	80	283.33
	3	19	76	146.50
	4	18	72	156.33
Wednesday	1	20	80	193.42
	2	20	80	1113.29
	3	20	80	118.28
	4	20	80	171.02
	5	15	60	67.77
Thursday	1	20	80	484.56
	2	20	80	820.51
	3	20	80	259.38
	4	17	68	144.31
	5	17	68	155.67
Friday	1	20	80	1109.08
	2	19	76	77.37
	3	19	76	145.21
	4	20	80	105.79
	5	20	80	222.39

6. CONCLUSIONS

In this thesis, we have studied VRPTD with asymmetric distances. The problem aims to determine efficient and effective tours. Requests of successive customers are satisfied by identical vehicles and each vehicle begins and ends its tour in the same depot. We propose two mathematical programming formulations for the problem. Since exact solutions cannot deal with large instances within a reasonable amount of time, we also develop a TS algorithm for solving the problem. This algorithm starts with feasible solutions generated by CW savings heuristic and employs seven types of move operators together as neighboring structures with LS algorithm.

To test the performance of proposed TS algorithm, we solve some of the CMT problems with Euclidean metric in literature by converting them to our problem and compare the results with best known solutions. The average gap between our algorithm and the best known solution is 3.96%. We also create a set of random asymmetric instances derived from the Solomon benchmark problems. These instances differ from our problem in the respect that they have earliest arrival time restriction and are symmetric distances. We relax this restriction, and create random asymmetric distance matrix with their coordinates. To test the TS performance, we also solve the instances by developed two mathematical models using CPLEX 12.5. We observe that the mathematical model based on single commodity flow is more efficient with respect to solution quality and computation time. We obtain optimal values for 36 of 42 generated asymmetric instances. When we compare the results, we can obtain optimal values for 28 of 42 instances by TS algorithm. The average gap of the rest is 1.40%. For some instances, we can handle the optimal values without using TS algorithm. However, there are instances which it does not improve the initial solution, too. Since proposed algorithm is constructed to deal with asymmetric distances, it requires more computational times for performing TS algorithm.

Our algorithm also deals with the cash transportation problem faced by a security carrier company in Turkey. Our model fits the company’s requirements. Based on the transportation information provided by the company, effective and efficient tours are sought to be determined by our algorithm.

In this study, we assume that travelling times are known and constant. However, in real life, the vehicle’s planned tour can be changed by a new customer request and vehicle speed is affected by heavy traffic congestion. Future attempts may be to incorporate this factor.

APPENDIX A: CPU TIMES FOR TEST INSTANCES

Table A.1. CPU times for generated asymmetric instances.

Problem	Number of Customer	CPU for CPLEX-Model I (s)	CPU for CPLEX-Model II (s)	CPU for CW+LS (s)	CPU for TS (s)
C203	5	0.2	0.3	0.2	668.9
	10	4.7	0.3	0.2	1176.7
	15	472.9	1.7	0.2	1534.7
	20	10800.0	5.9	0.3	5545.5
	25	10800.0	131.7	0.3	1089.9
	30	10800.0	9.1	0.3	5388.9
	50	2662.8	10800.5	0.4	14423.3
C204	5	0.5	0.3	0.2	576.4
	10	1.3	0.31	0.2	915.9
	15	259.3	1.7	0.2	2334.2
	20	5667.1	2.3	0.2	4265.4
	25	10800.0	5.1	0.3	3796.0
	30	10800.0	6.3	0.3	7213.8
	50	10800.0	630.9	0.4	10362.5
RC101	5	0.60	0.3	0.2	598.3
	10	734.6	0.7	0.2	1728.6
	15	179.4	2.8	0.2	3905.9
	20	10800.0	194.5	0.3	8576.6
	25	10800.3	6530.5	0.3	8587.6
	30	10800.3	4132.1	0.3	6228.0
	50	10800.3	10809.0	0.4	6172.8
RC108	5	0.06	0.3	0.2	566.3
	10	135.31	0.1	0.2	1579.9
	15	10800.0	0.3	0.2	3289.0
	20	10800.0	1.6	0.3	8369.6
	25	10800.1	2.8	0.3	11139.5
	30	10800.0	8.3	0.3	16407.1
	50	10800.0	10800.0	0.4	12532.1
RC202	5	0.3	0.1	0.2	451.3
	10	14.5	0.3	0.2	723.9
	15	10659.1	0.6	0.2	1210.2

Table A.1. – (cont.).

Problem	Number of Customer	CPU for CPLEX-Model I (s)	CPU for CPLEX-Model II (s)	CPU for CW+LS (s)	CPU for TS (s)
RC202	20	10800.0	10803.2	0.3	1901.3
	25	10800.0	3.3	0.3	1897.3
	30	10800.0	7.6	0.3	2283.5
	50	10800.0	10803.0	0.4	7541.6
RC206	5	0.08	0.2	0.2	452.1
	10	14.52	0.2	0.2	762.4
	15	10799.9	0.4	0.2	1056.9
	20	10800.0	0.7	0.3	1356.4
	25	10800.0	4.4	0.3	1985.4
	30	10800.0	6.5	0.3	4339.7
	50	10800.0	10801.1	0.4	14808.1

Table A.2. CPU times for CMT problems.

Problem	Number of Customers	CW+LS (s)	CPU (s)
C6	50	0.3	8484.4
C7	75	0.7	4085.9
C8	100	0.2	2956.1
C9	150	0.3	8252.0
C10	199	1.2	24330.5
C13	120	0.4	21947.6
C14	100	0.2	22031.5

REFERENCES

1. Dantzig, G. and J. Ramser, "The Truck Dispatching Problem," *Management Science*, Vol. 6, No. 1, pp. 80–91, 1959.

2. Lenstra, J. K. and A. H. G. R. Kan, "Complexity of Vehicle Routing Problem and Scheduling Problems," *Networks*, Vol. 11, No. 2, pp. 221–227, 1981.

3. Marinakis, Y. and A. Migdalas, "Annotated Bibliography in Vehicle Routing," *Operational Research*, Vol. 7, No. 1, pp. 27–46, 2007.

4. Solomon, M., "Algorithms for The Vehicle Routing and Scheduling Problems with Time Window Constraints," *Operations Research*, Vol. 35, No. 2, pp. 254–265, 1987.

5. Ahn, B. and J. Shin, "Vehicle-Routeing with Time Windows and Time-Varying Congestion," *Journal of the Operational Research Society*, Vol. 42, No. 5, pp. 393–400, 1991.

6. Desrochers, M., J. Desrosiers, and M. M. Solomon, "A New Optimization Algorithm for The Vehicle Routing Problem with Time Windows," *Operations Research*, Vol. 40, No. 2, pp. 342–354, 1992.

7. Thangiah, S., *Vehicle Routing with Time Windows Using Genetic Algorithms*, Vol. 3, No. 1. Citeseer, 1993, pp. 1–24.

8. Antes, J. and U. Derigs, "A New Parallel Tour Construction Algorithm for The Vehicle Routing Problem with Time Windows." 1995.

9. Potvin, J. Y., T. Kervahut, B.-L. Garcia, and J.-M. Rousseau, "The Vehicle Routing Problem with Time Windows Part I: Tabu Search," *INFORMS Journal on Computing*, Vol. 8, No. 2, pp. 158–164, 1996.

10. Potvin, J. and S. Bengio, "The Vehicle Routing Problem with Time Windows Part II: Genetic Search," *INFORMS Journal on Computing*, Vol. 8, No. 2, pp. 165–172, 1996.

11. Desaulniers, G., J. Lavigne, and F. Soumis, "Multi-Depot Vehicle Scheduling Problems with Time Windows and Waiting Costs," *European Journal of Operational Research*, Vol. 111, No. 3, pp. 479–494, 1998.

12. Liu, F. and S. Shen, "A Method for Vehicle Routing Problem with Multiple Vehicle Types and Time Windows," *Proceedings of the National Science Council Part A: Physical Science and Engineering*, Vol. 23, No. 4, pp. 526–536, 1999.

13. Cordone, R. and R. Calvo, "A Heuristic for The Vehicle Routing Problem with Time Windows," *Journal of Heuristics*, Vol. 7, No. 2, pp. 107–129, 2001.

14. Lee, L., K. Tan, K. Ou, and Y. Chew, "Vehicle Capacity Planning System: A Case Study on Vehicle Routing Problem with Time Windows," *IEEE Transactions on Systems, Man and Cybernetics: Part A (Systems and Humans)*, Vol. 33, No. 2, pp. 169–178, 2003.

15. Lau, H., M. Sim, and K. Teo, "Vehicle Routing Problem with Time Windows and A Limited Number of Vehicles," *European Journal of Operational Research*, Vol. 148, No. 3, pp. 559–569, 2003.

16. Cordeau, J. F., G. Laporte, and A. Mercier, "A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows," *Journal of the Operational Research Society*, Vol. 52, No. 8, pp. 928–936, 2001.

17. Cordeau, J. F., G. Laporte, and A. Mercier, "Improved Tabu Search Algorithm for The Handling of Route Duration Constraints in Vehicle Routing Problems with Time Windows," *Journal of the Operational Research*, Vol. 55, No. 5, pp. 542–546, 2004.

18. Bräysy, O., G. Hasle, and W. Dullaert, "A Multi-Start Local Search Algorithm for The Vehicle Routing Problem with Time Windows," *European Journal of Operational Research*, Vol. 159, No. 3, pp. 586–605, 2004.

19. Berger, J., M. Barkaoui, and O. Braysy, "A Route-Directed Hybrid Genetic Approach for The Vehicle Routing Problem with Time Windows," *Infor-Information Systems and Operational Research*, Vol. 41, No. 2, pp. 179, 2003.

20. Berger, J. and M. Barkaoui, "A Parallel Hybrid Genetic Algorithm for The Vehicle Routing Problem with Time Windows," *Computers & Operations Research*, Vol. 42, No. 3, pp. 415–431, 2004.

21. Homberger, J. and H. Gehring, "A Two-Phase Hybrid Metaheuristic for The Vehicle Routing Problem with Time Windows," *European Journal of Operational Research*, Vol. 162, No. 1, pp. 220–238, 2005.

22. Haghani, A. and S. Jung, "A Dynamic Vehicle Routing Problem with Time-Dependent Travel times," *Computers & Operations Research*, Vol. 32, No. 11, pp. 2959–2986, 2005.

23. Montemanni, R. and L. Gambardella, "Ant Colony System for A Dynamic Vehicle Routing Problem," *Journal of Combinatorial Optimization*, Vol. 10, No. 4, pp. 327–343, 2005.

24. Ombuki, B., B. Ross, and F. Hanshar, "Multi-Objective Genetic Algorithms for Vehicle Routing Problem with Time Windows," *Applied Intelligence*, Vol. 24, No. 1, pp. 17–30, 2006.

25. Repoussis, P. and D. Paraskevopoulos, "A Reactive Greedy Randomized Variable Neighborhood Tabu Search for The Vehicle Routing Problem with Time Windows," *Hybrid Metaheuristics Lecture Notes in Computer Science*, Vol. 35, No. 4, pp. 124–138, 2006.

26. Paraskevopoulos, D. and P. Repoussis, "A Reactive Variable Neighborhood Tabu Search for The Heterogeneous Fleet Vehicle Routing Problem with Time Windows," *Journal of Heuristics* Vol. 14, No. 5, pp. 425–455, 2008.

27. Kallehauge, B., J. Larsen, and O. Madsen, "Lagrangian Duality Applied to The Vehicle Routing Problem with Time Windows," *Computers & Operations Research*, Vol. 33, No. 5, pp. 1464–1487, 2006.

28. Fu, Z., R. Eglese, and L. Li, "A Unified Tabu Search Algorithm for Vehicle Routing Problems with Soft Time Windows," *Operational Research Society*, Vol. 59, No. 5, pp. 663–673, 2007.

29. Dell'Amico, M. and M. Monaci, "Heuristic Approaches for The Fleet Size and Mix Vehicle Routing Problem with Time Windows," *Transportation Science*, Vol. 41, No. 4, pp. 516–526, 2007.

30. Ostertag, A. and K. Doerner, "POPMUSIC for A Real-World Large-Scale Vehicle Routing Problem with Time Windows," *Journal of the Operational Research Society*, Vol. 60, No. 7, pp. 934–943, 2008.

31. Cheng, C. and K. Wang, "Solving A Vehicle Routing Problem with Time Windows by A Decomposition Technique and A Genetic Algorithm," *Expert Systems with Applications*, Vol. 36, No. 4, pp. 7758–7763, 2009.

32. Salani, M., "The Vehicle Routing Problem with Discrete Split Delivery and Time Windows Time Windows," in *9th Swiss Transport Research Conference, Ascona, Switzerland*, 2009.

33. Ho, S. C. and D. Haugland, "A Tabu Search Heuristic for The Vehicle Routing Problem with Time Windows and Split Deliveries," *Computers & Operations Research*, Vol. 31, No. 12, pp. 1947–1964, 2004.

34. Belfiore, P., H. Tsugunobu, and Y. Yoshizaki, "Scatter Search for Vehicle Routing Problem with Time Windows and Split Deliveries," *Vehicle Routing Problem*, pp. 1–15, 2008.

35. Ma, X., "Vehicle Routing Problem with Time Windows Based on Improved Ant Colony Algorithm," *2010 Second International Conference on Information Technology and Computer Science*, pp. 94–97, 2010.

36. Mao, Y. and Y. Deng, "Solving Vehicle Routing Problem with Time Windows with Hybrid Evolutionary Algorithm," *2010 Second WRI Global Congress on Intelligent Systems*, Vol. 1, pp. 335–339, 2010.

37. Carlos, J., H. Ali, R. Brito, and A. Silveira, "A Multi Objective Approach to Solve Capacitated Vehicle Routing Problems with Time Windows Using Mixed Integer Linear Programming," *International Journal of Advanced Science and Technology*, Vol. 28, pp. 1–8, 2011.

38. Potvin, J., Y. Xu, and I. Benyahia, "Vehicle Routing and Scheduling with Dynamic Travel Times," *Computers & Operations Research*, Vol. 38, No. 7, pp. 1086–1090, 2006.

39. Lorini, S., J.-Y. Potvin, and N. Zufferey, "Online Vehicle Routing and Scheduling with Dynamic Travel Times," *Computers & Operations Research*, Vol. 38, No. 7, pp. 1086–1090, 2011.

40. Gan, X., Y. Wang, S. Li, and B. Niu, "Vehicle Routing Problem with Time Windows and Simultaneous Delivery and Pick-Up Service Based on MCPSO," *Mathematical Problems in Engineering*, Vol. 2012, pp. 1–11, 2012.

41. Gong, Y., J. Zhang, and O. Liu, "Optimizing The Vehicle Routing Problem with Time Windows: A Discrete Particle Swarm Optimization Approach," *Systems, Man, Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions*, Vol. 42, No. 2, pp. 254–267, 2012.

42. Moon, I., J.-H. Lee, and J. Seong, "Vehicle Routing Problem with Time Windows Considering Overtime and Outsourcing Vehicles," *Expert Systems with Applications*, Vol. 39, No. 18, pp. 13202–13213, 2012.

43. Kritikos, M. N. and G. Ioannou, "The Heterogeneous Fleet Vehicle Routing Problem with Overloads and Time Windows," *International Journal of Production Economics*, No. 1984, pp. 1–8, 2013.

44. Vidal, T., T. G. Crainic, M. Gendreau, and C. Prins, "A Hybrid Genetic Algorithm with Adaptive Diversity Management for A Large Class of Vehicle Routing Problems with Time-Windows," *Computers & Operations Research*, Vol. 40, No. 1, pp. 475–489, 2013.

45. Baños, R., J. Ortega, C. Gil, A. Fernández, and F. de Toro, "A Simulated Annealing-based parallel multi-objective approach to vehicle routing problems with time windows," *Expert Systems with Applications*, Vol. 40, No. 5, pp. 1696–1707, 2013.

46. Thangiah, S. and I. Osman, "Algorithms for The Vehicle Routing Problems with Time Deadlines," American Journal of Mathematical and Management Sciences, Vol. 13, pp. 323–355, 1993.

47. Park, Y. B., "A Solution of The Bicriteria Vehicle Scheduling Problems with Time and Area-Dependent Travel Speeds," *Computers & Industrial Engineering*, Vol. 38, No. 1, pp. 173–187, 2000.

48. Park, Y., "A Hybrid Genetic Algorithm for The Vehicle Scheduling Problem with Due Times and Time Deadlines," *Int. J. Production Economics* , Vol. 73, 2001.

49. Kang, K. H., B. K. Lee, Y. H. Lee, and Y. H. Lee, "A Heuristic for The Vehicle Routing Problem with Due Times," *Computers & Industrial Engineering*, Vol. 54, No. 3, pp. 421–431, 2008.

50. Karlaftis, M. G., K. Kepaptsoglou, and E. Sambracos, "Containership Routing with Time Deadlines and Simultaneous Deliveries and Pick-ups," *Transportation Research Part E: Logistics and Transportation Review*, Vol. 45, No. 1, pp. 210–221, 2009.

51. Gavish, B. and S. C. Graves, "The Travelling Salesman Problem and Related Problems," *Operations Research Center, Massachusetts Institute of Technology*, p. GR–078–78, 1978.

52. Aras, N., D. Aksen, and M. Tuğrul Tekin, "Selective Multi-Depot Vehicle Routing Problem with Pricing," *Transportation Research Part C: Emerging Technologies*, Vol. 19, No. 5, pp. 866–884, 2011.

53. Aksen, D., O. Kaya, F. S. Salman, and Y. Akça, "Selective and Periodic Inventory Routing Problem for Waste Vegetable Oil Collection," *Optimization Letters*, Vol. 6, No. 6, pp. 1063–1080, 2012.

54. Clarke, G. and J. Wright, "Scheduling of Vehicles From A Central Depot to A Number of Delivery Points," *Operations research*, Vol. 12, No. 4, pp. 568–661, 1964.

55. Altınel, İ. K. and T. Öncan, "A New Enhancement of The Clarke and Wright Savings Heuristic for The Capacitated Vehicle Routing Problem," *Journal of The Operational Research Society*, Vol. 56, No. 8, pp. 954–961, 2004.

56. Toth, P. and D. Vigo, "The Granular Tabu Search and Its Application to The Vehicle-Routing Problem," *INFORMS Journal on Computing*, Vol. 15, No. 4, pp. 333–346, 2003.

57. Groër, C., B. Golden, and E. Wasil, "A Library of Local Search Heuristics for The Vehicle Routing Problem," *Mathematical Programming Computation*, Vol. 2, No. 2, pp. 79–101, 2010.

58. Aksen, D., Z. Özyurt, and N. Aras, "The Open Vehicle Routing Problem with Driver Nodes and Time Deadlines," *Journal of The Operational Research Society*, Vol. 58, No. 9, pp. 1223–1234, 2007.

59. Cordeau, J. F., M. Gendreau, and G. Laporte, "A Tabu Search Heuristic For Periodic and Multi-Depot Vehicle Routing Problems," *Networks*, Vol. 30, No. 2, pp. 105–119, 1997.

60. Neo, "*The VRP Web*", 2007, http://neo.lcc.uma.es/radi-aeb/WebVRP/, June 2013

61. Solomon, M., "*VRPTW Benchmark Problem*", 2005, http://web.cba.neu.edu/~msolomon/problems.htm, June 2013