Delft University of Technology
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft Institute of Applied Mathematics

A Solution Method for Vehicle Routing Problems with Time-Dependent Travel Times

A thesis submitted to the
Delft Institute of Applied Mathematics
in partial fulfillment of the requirements

for the degree

MASTER OF SCIENCE
in
APPLIED MATHEMATICS

by

ESTER STEGERS

Delft, the Netherlands
December 11, 2009

MSc THESIS APPLIED MATHEMATICS

"A Solution Method for Vehicle Routing Problems with Time-Dependent Travel Times"

ESTER STEGERS

Delft University of Technology

Daily supervisors

Dr. H.J. Quak

Ir. C.J.L Rothkrantz

Responsible professor

Prof. dr.ir. K.I. Aardal

Other thesis committee member

Dr. R.J. Fokkink

December 11, 2009

Delft, the Netherlands

Abstract

The goal of this thesis is developing an efficient method that produces good quality solutions to real-life Vehicle Routing Problems with time-dependent travel times. The Time-Dependent Vehicle Routing Problem (TDVRP) is one of the most challenging combinatorial optimisation problems and belongs to the category of NP-hard problems. Therefore, several heuristic algorithms are explored and the most promising that handles the TDVRP is implemented.

This study is motivated by the effect of dynamic changes in traffic, such as rush hour traffic jams, and the resulting logistics cost. Due to this increasing effect TNO would like to include this in their advice. Currently, TNO uses the logistics optimisation model RESPONSE^{TM} to give advice to logistics companies. However, this tool is based on an Evolution Strategy that cannot handle time-dependent travel times. Therefore, in this thesis, I search for possibilities to improve this. A Construction and Route Improvement Algorithm (CRIA) is developed that deals with time-dependent travel times. The routes must be designed in such a way that first the total number of vehicles are minimised and secondly the total route durations are minimised.

The developed CRIA takes a route improvement approach instead of the local improvement approach that most researchers take, and is integrated with a time-dependent shortest path algorithm that will be executed during the TDVRP process. Due to varying travel times, paths (with different distances) may result in a shorter path (in time) between customers at different times of the day. Using the time-dependent shortest path algorithm, also developed in this thesis, these shorter paths may be found. In practise, this approach allows for the creation of routes that avoid congested areas during rush hours. No other published work using a similar algorithm is known to exist.

Notice that in an urban context, variable traffic conditions play an essential role and cannot be ignored in order to derive realistic solutions to routing problems. In this thesis it is shown that when dealing with time dependent scenarios, the solutions generated by a non time-dependent algorithm may become infeasible. Moreover there is a possibility to find solutions that are better than the solutions found with constant speed applied to the time-dependent context.

The method is tested on one of the Solomon's benchmark problems and on a real-life dataset. The developed algorithm seems to generate solutions of good quality. The computational complexity of the solution method is analysed and experimental results indicate that the computation time increases proportionally to the number of time-zones, as well as to the square of the number of customers.

When time-dependent measurement data connected to a geographic network become available, the services of TNO can play an even more important role for the logistics planning of companies.

Key words: *Vehicle Routing Problems, time dependent travel time and speed, congestion, heuristic methods, transportation, logistics cost.*

Acknowledgements

This thesis has been written for obtaining the degree of Master of Science in Applied Mathematics at the faculty of Electrical Engineering, Mathematics and Computer Sciences of Delft University of Technology. The actual thesis research has been carried out at TNO Business Unit Mobility and Logistics in Delft.

First of all I would like to thank TNO for giving me the opportunity to perform my thesis research there. It was a pleasant and informative working time. I much appreciate that I was really involved in the team Logistics of TNO BU M&L. The team meetings were interesting and the informal parties too.

I would like to thank my daily supervisors at TNO, Cyril Rothkrantz and Hans Quak, for supporting me during the project, and for reading and correcting my thesis. Cyril has given me the opportunity to do research on a topic of his own personal interest. Due to his many obligations at TNO he could however not devote enough time to do this research on his own. His enthusiasm, interest and knowledge of the topic inspired me, and it was nice to work in his presence and under his guidance. He taught me a lot and especially he introduced me to the Visual Basic code in which the complex logistics optimisation model, RESPONSE™ is written. In addition, Hans has supported me in the planning, such that I was focussed on the primary tasks and never felt lost.

Next, I would like to thank my colleagues of TNO who supported me, especially Diederik de Ree for teaching me a lot about efficient programming. Next to communicating his knowledge, his social talks amused me. Moreover I will never forget the 3 o'clock coffee breaks with the other Master students, Yvo, Robbert and Silvie.

I would also like to thank my supervisor at TU Delft, Karen Aardal. I want to thank her for reading and correcting my thesis and for her inspiring enthusiasm during my research. Next, I want to thank Robbert Fokkink to take part in the examination committee.

Last but not least, I want to thank my family for believing in me and supporting me during my thesis research and study. Especially I would like to thank Wouter for being sweet during some stressful moments and helping me with writing my thesis in correct English. Without all this help I would not have been able to finish my thesis and conclude my study at Delft University of Technology.

Delft, University of Technology — Ester Stegers
December 11, 2009

Table of Contents

List of Figures

List of Tables

Acronyms

ACO Ant Colony Optimisation

ACVRP Asymmetric Capacitated Vehicle Routing Problem

BU M&L Business Unit Mobility and Logistics

CA Construction Algorithm

CRIA Construction and Route Improvement Algorithm

CVRP Capacitated Vehicle Routing Problem

DVRP Dynamic Vehicle Routing Problem

EA Evolutionary Algorithm

ES Evolution Strategy

FIFO First-In-First-Out

FLM Facility Location Model

GA Genetic Algorithm

GRASP Greedy Randomised Adaptive Search Procedure

GVRP Generalised Vehicle Routing Problem

HVRP Heterogeneous Vehicle Routing Problem

MDVRP Multiple Depot Vehicle Routing Problem

NP Non-deterministic Polynomial-time

SA Simulated Annealing

SVRP Stochastic Vehicle Routing Problem

TDVRP Time-Dependent Vehicle Routing Problem

TS Tabu Search

TSP Travelling Salesman Problem

VB Visual Basic

VNS Variable Neighbourhood Search

VRP Vehicle Routing Problem

VRPM Vehicle Routing Problem with Multiple routes per vehicle

VRPPD Vehicle Routing Problem with Pick-up and Delivery

VRPTW Vehicle Routing Problem with Time Windows

Chapter 1

Introduction

One of the current challenges in logistics optimisation is the high degree of dynamics and uncertainty that influences so many transportation companies. The Vehicle Routing Problem (VRP) plays a central role in this field. TNO Business Unit Mobility and Logistics (BU M&L) has designed a logistics optimisation model to handle VRPs, called RESPONSE™. Currently this model does not explicitly take into account changes and uncertainty in travel times due to traffic conditions. It is very interesting to extend RESPONSE™ with possibilities for taking these dynamics and uncertainties into account in such a way that it can handle time-dependent travel times. This dynamic and therefore more realistic planning in logistics gives TNO BU M&L the opportunity to bring out a broader scale of advice supported by RESPONSE™.

1-1 Problem Investigation

Congestion creates a substantial variation in travel speeds during morning and evening rush hours. This is problematic for all vehicle routing models that rely on a constant value to represent vehicle speeds. Urban route designs that ignore these significant speed variations result in inefficient, unrealistic and suboptimal solutions. These poorly designed routes that lead freight vehicles into congested urban traffic, result in some cases in a waste of valuable time, and customers have to wait unreasonably long without having any reliable information about the actual arrival times of the vehicles. In these circumstances, it becomes difficult to satisfy the time windows during which the customers must be visited. This increases supply chain and logistics cost.

TNO BU M&L has observed these dynamic changes and is including this effect in their models. In order to obtain a more realistic solution, a method should be created that deals with time-dependent travel times. Van Rijswijck (2008) reported on the need for complex optimisation algorithms and the need for time dependent measurement data. This thesis focuses on the first. Not only does the optimisation algorithm need to calculate a low cost feasible solution, it also has to do this efficiently. Efficiency becomes critical for the large problems often occurring in a logistics company. Note that basic assumptions for constructing an algorithm that can handle time-dependent travel times are that all input information (like service requests and all travel times over all road types) are known before the start of the planning horizon.

1-2 Problem Statement and Objective

The objective of this master's thesis can be defined as follows:

> Developing an efficient method that produces good quality solutions to real-life Vehicle Routing Problems with time-dependent travel times.

In this thesis, this research objective is dealt as follows: first a literature review with regard to solution methods for solving vehicle routing problems with time-dependent travel times is performed. In parallel, an analysis of the current vehicle routing model (with constant travel times on each path) is done by studying the work of Uil (2004). From this research, interesting algorithms are explored and the most promising solution method is implemented and tested. In this way an algorithm that is able to handle time-dependency is designed and implemented. Next, the developed Construction and Route Improvement Algorithm is tested, in terms of solution quality as well as computation time, on a small benchmark problem as well on a real-life problem.

1-3 Profile TNO

This thesis is written during an internship at TNO BU M&L, that is part of the core area Built Environment and Geosciences. TNO is a research organisation with the primary goal to apply scientific knowledge in practice. TNO has about 4,500 employees, the head office is established in Delft and there are a lot of other locations stretched out over the Netherlands. TNO has also a number of small foreign offices in countries like Belgium, China, Korea, France, Japan and the USA.

TNO has five core areas in total, each of which offers its own package of products and services. Topical issues of the core area Built Environment and Geosciences include the use of the subsurface, mobility, the infrastructure, the renewal of the inner cities and the formation of a network between cities. Vlaar (2009) presents more information about the organisation TNO.

In team Logistics of TNO BU M&L, the logistics experts apply scientific knowledge to logistics with the aim of helping the logistics to companies to grow with the increasing dynamics and complexity of trade and production processes. In addition, smart logistics contribute to achieving social goals such as reducing traffic jams and emissions as well as boosting traffic safety. The logistics experts work for large and medium-sized companies, intermediary organisations and authorities in both the Netherlands and Europe. Team Logistics of TNO offers services like:

- optimising and designing logistics networks using a model-based approach;
- evaluating and preparating of an effective policy for logistics and freight transport;
- improving cooperation in logistics networks;
- creating innovative logistics solutions;
- advising about resilience and risk management in logistics.

Verweij (2009) presents more information about these services.

1-4 Thesis Outline

This thesis is organised as follows: Chapter 2 introduces mathematical notation and formulates the Vehicle Routing Problem (VRP) followed by a literature review for the Time-Dependent Vehicle Routing Problem (TDVRP). An overview of the methodology used in RESPONSE™ is presented in Chapter 3. Chapter 4 presents the Construction and Route Improvement Algorithm (CRIA) that can handle time-dependent travel times, developed during this master's thesis. Here, results in terms of solution quality as well as computation time are presented and discussed. In Chapter 5 a realistic view is made by introducing a real-life dataset on which the CRIA is tested. Chapter 6 ends this thesis with conclusions and recommendations. Appendices, bibliography and an index are included at the end, and a detailed list of the acronyms employed in this thesis is added after the table of contents.

Chapter 2

Vehicle Routing Problems

This chapter presents a global overview of the Vehicle Routing Problem (VRP). First, Section 2-1 gives a mathematical formulation as well as solution methods for the basic VRP and describes several VRP variants. Next, Section 2-2 defines the TDVRP with their developed solution methods.

2-1 Vehicle Routing Problems and their Solutions

The VRP is one of the most challenging combinatorial optimisation problems. It consists of designing the optimal set of routes for a fleet of vehicles parked at a central depot in order to service a given set of customers with a fixed demand. The VRP originated from the Travelling Salesman Problem (TSP), a special case of the VRP in which only one vehicle with 'sufficient' capacity is available. The TSP is the simplest, and probably most famous of routing problems known to most researchers. The interest in the VRP is motivated by its practical relevance as well as its considerable mathematical difficulty. In some market sectors, transportation cost form a large part of the total cost of a product. Therefore, the utilisation of computerised methods for transportation often results in significant savings ranging from 5% to 20% of the total product cost, as reported by Toth and Vigo (2001). Since the VRP was introduced 50 years ago by Dantzig and Ramser (1959), a lot of papers are devoted to the exact and approximate (near-optimal) solution of the many variants of this problem. Subsection 2-1-1 gives a mathematical formulation of the basic VRP. Solution methods for the basic VRP are discussed in Subsection 2-1-2 and several variants of the VRP are described in Subsection 2-1-3.

2-1-1 Mathematical Formulation

To obtain a mathematical formulation of the basic VRP, first some notation and definitions are needed. This thesis uses the same formulation as the one found in Baldacci *et al.* (2008a).

Let $G = (V, A)$ be a directed graph with vertex set $V = \{0, \ldots, n\}$ and arc set A. Then the input data becomes

- vertex 0 corresponds to the depot;
- vertices $1, \ldots, n$ correspond to the customers, take $V' = \{1, \ldots, n\}$;
- a set of $M = \{1, \ldots, m\}$ identical vehicles (a homogeneous fleet), each vehicle with capacity C, is available at the depot;

- the vehicles must return to the depot they originated from;
- each customer $i \in V'$ is associated with a known demand $d_i \geq 0$ to be delivered (assume $d_0 = 0$ and $d_i \leq C$ for all $i \in V'$);
- a fixed cost F for each vehicle required that models the rental or capital amortisation cost of a vehicle;
- a non-negative cost c_{ij} is associated with each arc $(i, j) \in A$ representing the travel cost between vertices i and j (assume $c_{ii} = 0$).

In addition, the decision variables become

- the binary variables x_{ij} are used as vehicle flow variables that take value 1 if a vehicle travels directly from customer i to customer j, and 0 otherwise;
- the flow variables y_{ij} specify the quantity of goods that a vehicle carries when it leaves customer i to service customer j.

With this notation, the mathematical formulation of the VRP becomes

minimise:

$$\sum_{j \in V'} F x_{0j} + \sum_{i,j \in V,\ i \neq j} c_{ij} x_{ij}, \tag{2-1}$$

subject to:

$$\sum_{i \in V} x_{ij} = 1, \qquad \forall j \in V' \tag{2-2}$$

$$\sum_{i \in V} x_{ip} - \sum_{j \in V} x_{pj} = 0 \qquad \forall p \in V' \tag{2-3}$$

$$\sum_{j \in V'} x_{0j} \leq m \tag{2-4}$$

$$\sum_{i \in V} y_{ij} - \sum_{i \in V} y_{ji} = d_j \qquad \forall j \in V' \tag{2-5}$$

$$d_j x_{ij} \leq y_{ij} \leq (C - d_i) x_{ij} \qquad \forall i, j \in V, i \neq j \tag{2-6}$$

$$y_{ij} \geq 0 \qquad \forall i, j \in V, i \neq j \tag{2-7}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in V, i \neq j. \tag{2-8}$$

In the above formulation, the objective (2-1) is to find a collection of simple circuits in the graph G (each circuit corresponding to a vehicle route) with minimal cost (vehicle cost and travel cost) such that:

- each customer is visited once by exactly one vehicle route (constraints 2-2);
- if a vehicle visits a customer, it must also depart from it (constraints 2-3);
- the number of vehicle routes is not greater than the number of available vehicles, i.e., there is a finite vehicle fleet (constraints 2-4);
- the difference between the quantity of goods a vehicle carries before and after visiting a customer is equal to the demand of that customer (constraints 2-5);
- the sum of the demands of the customers visited by each vehicle route does not exceed the given vehicle capacity C (constraints 2-6).

2-1-2 Solution Methods

The VRP belongs to the category of NP-hard problems, meaning that the computational effort required to solve this problem increases exponentially with the problem size in the worst case. A small VRP (up to 100 customers) is solvable with exact algorithms, but for a large VRP (more than 100 customers) it is desirable to obtain approximate solutions using heuristics as well as metaheuristics. These methods often give routes of sufficient quality within reasonable computation time when exact algorithms are not fast enough. More detailed information about exact approaches, heuristic methods and metaheuristics is given below in the same order used by Dorronsoro (2007).

Exact Approaches

For small problems, exact approaches are proposed that evaluate implicitly, every possible solution to obtain the best solution. A well-known exact method is the branch and bound method, which consists of a systematic implicit enumeration of all feasible solutions. Using lower and upper bounds on the optimal objective value, more and more subsets of the feasible solutions will be rejected, such that the optimal solution appears. Another exact approach is the branch and cut method, a hybrid of the branch and bound method and the cutting plane method. The cutting plane method adds linear inequalities, called cuts, to the problem in order to define an as small as possible feasible set of the objective values. To prevent a slow convergence to the optimal value, the structure of the problem can be used to generate very good cuts. One of the most efficient exact methods in the literature is proposed by Baldacci *et al.* (2008b), who combine three different bounding procedures to compute the optimal solution.

Heuristic methods

To approximately solve larger problems within reasonable computation time, heuristic methods are used. Heuristic methods perform a relatively limited exploration of the search space and typically produce good quality solutions within modest computation times. The heuristic methods can be divided into two-phase methods and constructive methods. Two-phase methods are decomposed into two natural components, on the one hand clustering of customers into feasible routes and on the other hand actual route construction (like solving a Travelling Salesman Problem (TSP)), with possible feedback loops between the two stages. Constructive methods consist of route construction heuristics and local improvement heuristics. A route construction heuristic quickly builds a feasible solution, but usually not the optimal one. The most well-known route construction heuristic is the greedy nearest neighbour heuristic. This heuristic starts at an arbitrary customer, subsequently it chooses the nearest customer as the next one to visit and so on, until a feasible solution is obtained. Another well-known route construction heuristic is the Clarke-Wright savings heuristic. This savings heuristic starts with an initial allocation of each customer to a separate route. Then for each pair of customers the cost savings of joining those customers on one route are calculated. Based on the values of these savings, the customers are sequentially joined into routes starting with the customer combination yielding the largest cost savings until no further savings can be achieved. Local improvement heuristics are heuristics based on simple route modifications. The most well-known local improvement heuristic is the k-opt heuristic, which removes k arcs and reconnects the routes in another possible way to find a better solution. Other well-known local improvement heuristic are the Or-opt heuristic (an intermediate heuristic between 2-opt and 3-opt), the crossover heuristic (two routes are mixed at one point), the customer relocation heuristic (a customer located at one route is changed to another one) and the customer exchange heuristic (two customers of two different routes are interchanged between the two routes).

Metaheuristics

Metaheuristics are more generic solution schemes that combine several heuristic methods. In these methods, the emphasis lies on performing a deep exploration of the most promising regions of the solution space. The quality of solutions produced by these methods has the potential to be higher than that obtained by only one heuristic method. A metaheuristic is a search strategy that first constructs an initial solution using a route construction heuristic. The initial solution is then improved by an iterated local search procedure, with phases of intensification and diversification. Intensification is achieved through finding an optimum in the neighbourhood of the initial solution using local improvement heuristics. When a local optimum has been found, several diversification mechanisms like a random restart, or penalising moves, may be invoked to 'jump' to an unexplored part of the search space in order to escape from poor local minima. More detailed information about local search is formulated by Aarts and Lenstra (1997).

There are several types of metaheuristics like Greedy Randomised Adaptive Search Procedure (GRASP), Ant Colony Optimisation (ACO),Variable Neighbourhood Search (VNS), Simulated Annealing (SA), Tabu Search (TS) and Evolutionary Algorithm (EA). An overview of these types is given below and more information including references on specific metaheuristics can be found in Gendreau *et al.* (2008).

Greedy Randomised Adaptive Search Procedure (GRASP) focuses on the route construction heuristic. GRASP uses an randomised greedy route construction heuristic within a multi start procedure to generate a variety of solutions. At each step of the route construction heuristic, customers with minimal added cost are kept in a restricted candidate list. Subsequently one customer is randomly chosen from that list and becomes the next customer to visit. After the construction process is completed, the solution is further improved with an iterated local search procedure.

Ant Colony Optimisation (ACO) is another metaheuristic with focus on the route construction. ACO uses 'pheromone', which represents the memory of the system in which good solutions are remembered. The basic idea comes from the communication and cooperation mechanisms among ants that allow them to find short paths from their nest to food sources. The communication medium is pheromone, a chemical compound that is laid down on the ground. While an isolated ant would travel randomly, an ant detecting a pheromone path will follow it, with some probability, and will strengthen it with its own pheromone. Based on this idea, the ACO is implemented associating 'pheromone' weights with each arc, which are related to the presence of the arc in good solutions previously constructed by the 'ants', i.e., each 'ant' uses a positive feedback mechanism to reinforce those arcs that belong to a good solution. After constructing the route using pheromone, the solution is further improved with an iterated local search procedure.

Variable Neighbourhood Search (VNS) is a metaheuristic with focus on the iterated local search procedure, which exploits neighbourhoods to escape from local minima. The first step is to select a set of neighbourhood structures in a random order. Start with an initial solution and take this as the current solution. Then a solution is generated in the first neighbourhood of the current solution, on which local improvement heuristics are applied. If the local minimum obtained is not better than the current solution, then the procedure is repeated by generating a solution in the next neighbourhood of the current solution. The search restarts from the first neighbourhood when either a solution that is better than the current solution is found or all neighbourhoods have been tried. This is repeated until the current solution cannot be improved anymore.

Simulated Annealing (SA) is another metaheuristic with focus on the iterated local search procedure, which allows bad solutions to escape from local minima. The basic idea comes from a physics process of annealing in which a pure lattice structure of a solid is made by increasing its temperature in a heat bath until it melts, then cooling it down slowly until it solidifies into a low-energy state. Based on this process in SA, a solution corresponds to a state and the solution cost to its energy. At each iteration the current solution is modified. A modification leading to a better solution is always accepted, but also a modification leading to increase in solution cost could be accepted. For the acceptance of solution cost increase, a probability criterion is defined in which modifications are more likely to be accepted if the 'temperature' parameter is high and the cost increase is low. When the temperature is sufficiently low, only improving modifications can be accepted and the method stops in a local minimum.

Tabu Search (TS) is like VNS and SA another metaheuristic with focus on the iterated local search procedure. The key idea is to forbid certain possible solutions based on a short-term memory, known as the tabu list. This list stores recently visited solutions to avoid short-term cycling. TS also allows bad solutions to escape from local minima.

Evolutionary Algorithm (EA) uses techniques inspired by Darwin's principle of natural selection such as recombination (also known as genetic crossover), mutation, selection (also known as survival of the fittest). The most popular type of EA is Genetic Algorithm (GA). This metaheuristic contains 4 processes. GA starts by generating multiple solutions often encoded as bit or integer strings, known as 'chromosomes'. Every solution is feasible, but does not necessarily have to be a good solution to the problem. This set of solutions is called a population and a single solution is called an individual. The second process of GA is the recombination. This process combines several individuals from the parent population to create new individuals. To create one new individual mostly two parents are selected and their most desirable features are combined in order to create an offspring solution, i.e., the new individual. To increase the chance of high-valued offspring in this recombination process, more offspring are generated than the number of individuals in the parent population. After the new population is generated, the third process starts, the mutation. Each individual is subjected to mutation, which may allow for escaping local minima by preventing the population of chromosomes from becoming too similar to each other. The result after the first three processes is the final offspring population. The fourth and final process of GA concerns the selection of a new starting population and saving the current best solution. Every offspring individual is rated by one or more criteria and the best or most promising individuals are promoted to a new parent population. To get a final solution with a high potential to have good quality, more iterations can be done in which processes two to four can be repeated. Another type of EA is Evolution Strategy (ES), mentioned by Homberger and Gehring (1999). According to them, differences between ES and GA lie in the representation of the problem (ES operates with non-encoded individuals) and in the mutation process. The mutation process from ES has a superior role over the recombination process. Where GA uses mutation for escaping local minima, ES also uses mutation for improvement of the solution, i.e., application of local improvement heuristics.

2-1-3 VRP Variants

In Subsection 2-1-1 a mathematical formulation of the basic VRP was given. This formulation makes a number of assumptions, including utilising capacity constraints, a single depot, a homogeneous fleet, one route per vehicle, etc. These assumptions can be eliminated by introducing additional constraints or relaxing certain constraints of the problem. This leads to several VRP variants. An overview of these variants is given in this subsection. It must be noted that several variants can be combined into one specific VRP.

Capacitated VRP

The Capacitated Vehicle Routing Problem (CVRP) is a VRP in which the emphasis is on the capacity constraints (in contrast to the TSP). Generally the CVRP is defined with a symmetric travel cost matrix, i.e., the travel cost c_{ij} from customer i to j equal the travel cost c_{ji} from customer j to i. The CVRP with an asymmetric travel cost matrix is explicitly denoted by the Asymmetric Capacitated Vehicle Routing Problem (ACVRP). In this case, the travel cost from customer i to j are not necessary equal to the travel cost from customer j to i.

VRP with Time Windows

One of the most important extensions of the basic VRP is the Vehicle Routing Problem with Time Windows (VRPTW). This variant introduces the additional restriction that a time window is associated with each customer, defining an interval in which arriving at the customer is allowed. Moreover there is a time window at the depot, which guarantees that each route must start and end within the time window associated with the depot.

The time windows are either soft or hard. A hard time window has a strict lower bound and upper bound, i.e., if a vehicle arrives before the lower bound of the customer time window an additional waiting time on the route is taken into account and if a vehicle arrives after the upper bound of the customer time window a solution becomes infeasible. A soft time window also adds a waiting time when a vehicle arrives too early at a customer, but when a vehicle arrives too late the solution remains feasible, however the total travel cost will be penalised by some amount.

Because the VRPTW contains waiting time, an extension of the objective of the basic VRP is needed. The objective of the VRPTW is not only to minimise the number of vehicles required (vehicle cost) and the total travel distance (travel cost), but also the total travel time incurred by the fleet of vehicles (travel time cost). Detailed research of the VRPTW is done by Bräysy and Gendreau (2005).

VRP with Pick-up and Delivery

The Vehicle Routing Problem with Pick-up and Delivery (VRPPD) is a VRP in which the possibility that customers return some products is admitted. All delivery demands start from the depot like in the basic VRP and all pick-up demand should be returned to the depot. No interchanges of products between customers are allowed. Using one depot, this procedure is called single sourcing VRPPD. An important restriction of the VRPPD is that the products that customers return to the deliver vehicle must fit, i.e., the vehicle capacity may never be exceeded.

Multiple Depot VRP

In contrast to the standard VRP with a single depot, the Multiple Depot Vehicle Routing Problem (MDVRP) is a VRP in which a company has several depots from which it can service its customers. A fleet of vehicles is parked at each depot and each vehicle has to return to its own depot. To handle realistic size MDVRP instances, the problem has to be split up in several single depot VRPs, which can be approximately solved separately. To split up the MDVRP, an assignment of customers to depots is required. For this, it is important to know how the depots and customers are positioned.

Heterogeneous VRP

The Heterogeneous Vehicle Routing Problem (HVRP) is a VRP with a heterogeneous fleet of vehicles, i.e., a fleet with multiple vehicle types. In the basic VRP a homogeneous fleet (a fleet

with identical vehicles) is used. In a homogeneous fleet, each vehicle has the same capacity C and the same fixed vehicle cost F. In a heterogeneous fleet, each vehicle type k has its own capacity C_k and is also associated with its own fixed vehicle cost equal to F_k. In addition, the travel cost between two vertices (between customers or customer-depot) is also dependent on the vehicle type in a heterogeneous fleet.

VRP with Multiple routes per vehicle

The Vehicle Routing Problem with Multiple routes per vehicle (VRPM) permits vehicles to make multiple routes to and from a depot within a given time period. In the basic VRP the assumption is made that each route from depot to depot is done by a different vehicle. The VRPM sets several routes after each other (if the time period allows it), which can be serviced by one vehicle. In this way, the minimal number of vehicles can be reduced, which gives a smaller minimal cost than the basic VRP.

Generalised VRP

The Generalised Vehicle Routing Problem (GVRP) is an extension of the VRP in which the vertex set V is partitioned into $n+1$ non-empty subsets (or clusters) $C_0, C_1, \ldots C_n$, with n the number of clusters to visit. Cluster C_0 has only one vertex (vertex 0 that corresponds to the depot), while each cluster $C_h \in V$ represents $r(h)$ possible locations of a customer, i.e., C_h has $r(h)$ vertices. The GVRP has the condition to visit each cluster C_h exactly once. A cluster can be interpreted as a neighbourhood with the possibility to drop off the product for this neighbourhood at the optimal place in this neighbourhood.

Time-Dependent VRP

In the Time-Dependent Vehicle Routing Problem (TDVRP), the travel time between two vertices of the network depends on the distance between the vertices and the time-dependent speed between these vertices. This in contrast to the basic VRP in which only the travel time depends on the distance and the directed fixed speed between the vertices. Due to the time-dependent travel times, the shortest path between two customers also becomes time-dependent. In the TDVRP, the day is divided into several time-zones, each having its own travel speeds over the network. An important property for the TDVRP is the First-In-First-Out (FIFO) property, also called the 'non-passing' property: If two vehicles leave from the same location for the same destination, travelling the same route, the one that leaves first will always arrive first, no matter how speed changes on the links during the travel. This property is important because it precludes the solution from inducing useless waiting times. Without the FIFO property, it can happen that a vehicle could wait at some location for the time when speeds are higher and then arrive at the desired location before another vehicle that had left before.

Stochastic VRP

The Stochastic Vehicle Routing Problem (SVRP) differs from the basic VRP by the introduction of some elements of variability, like random travel times, service times and demands. These elements are stochastic and one makes assumptions on the probability distribution of these elements. The basic VRP is termed static, where all information on the problem is known with good enough certainty and it is available before problem solving starts and does not change thereafter. In the case of SVRP, the information is also known a priori, but some elements are determined by predicting future events with taking uncertainties into account. This uncertainties can be traffic conditions that may severely influence driving time, variation in service time and imprecise information on customer demands.

Dynamic VRP

In the Dynamic Vehicle Routing Problem (DVRP), new information on the problem becomes available during plan execution. Examples of this new information can be new customer orders, changes in travel times due to unpredictable traffic conditions, vehicle breakdowns, early arrival and a change of demands. In the case of new customer orders, a set of orders is known in advance and a first schedule is calculated for it. If a new order dynamically arrives when the vehicles have already started executing their tours, the orders have to be planned again in order to include these new orders. The DVRP is sometimes also referred to as 'on-line' or 'real-time' transportation planning. Technological developments, such as wireless communication and global positioning systems have enabled the realisation of tools for the DVRP. An interesting overview of papers on the DVRP is given by Gendreau *et al.* (2008).

2-2 Time-Dependency in Vehicle Routing Problems

Because the objective of this thesis is developing an efficient method that produces good quality solutions to real-life Vehicle Routing Problems with time-dependent travel times, this section devotes more attention to this VRP variant. An overview of the best known results up to now are presented in Subsection 2-2-1. Moreover, Subsection 2-2-2 gives a short conclusion of this literature review.

2-2-1 Literature Review

The TDVRP is first formulated by Malandraki and Daskin (1992). They use a GRASP metaheuristic without further improvement by an iterated local search procedure. They handle small TDVRPs (with 10-25 customers and 2-3 time-zones) without time windows, in which the travel times over the network per time-zone are given (no shortest path calculations are needed). On each arc (between customers or customer-depot) a step function distribution of the travel time is introduced. In addition, research is done into time-dependent TSP.

Ichoua *et al.* (2003) were the first who introduced the FIFO property. They show that models based on discrete travel time does not satisfy the FIFO property. They set up a model based on discrete travel speed, i.e., they use a model that assigns a step function to the travel speed for each link, resulting in a piecewise linear travel time function that satisfies the FIFO property. They propose a solution methodology to approximately solve the TDVRP with soft time windows. The optimisation is done with a TS metaheuristic that handles the TDVRP with 100 customers and 3 time-zones. The model is also formulated for a dynamic environment, where not all service requests are known before the start of the planning horizon. Notice that they do not deal with time-dependent shortest path calculations, but they do experiments with fixed distances between customers.

Subsequently, Fleischmann *et al.* (2004) analyse constructive methods (including local improvement heuristics) to approximately solve the TDVRP with and without hard time windows. Their developed algorithms are tested on instances created from a traffic information system in the city of Berlin. They formulate travel time matrices, each travel time matrix corresponds to a specific time-zone and contains all travel times from a vertex i to a vertex j travelled in this specific time-zone. Neighbouring time-zones with similar travel times are joined to reduce memory requirements. To ensure the FIFO property on travel times, they use a smoothed travel time function that depends on departure time and time-zone bounds. Because this travel time function can be evaluated with constant effort, the limitation for the number of time-zones comes rather from the required storage space for the travel time matrices than from the computation

time. They can handle problems up to 700 customers with 10 time-zones within reasonable computation time as well as problems up to 100 customers with 50 time-zones.

Next, Haghani and Jung (2005) propose to use a GA metaheuristic to handle the TDVRP with soft time windows and a heterogeneous fleet of vehicles. They prefer a GA because of its significant advantage in dealing with complex constraints (that included time windows, multiple depots and a continuous travel time function) that can be embedded in the model. Moreover no hard to obtain parameters are needed for the implementation. The performance of the GA is evaluated by comparing its results with exact solutions and lower bounds. For small problems (with up to 10 customers and 15 time-zones), the GA provides almost the same results as the exact solutions, while its computation time is less than 10% of the time required to produce the exact solutions. For problems with up to 30 customers and 30 time-zones, the GA results have less than 8% gap with lower bounds. When the number of customers increases, the population for the GA increases accordingly because it is directly proportional to the number of customers. In order to handle problems larger than 70 customers in reasonable computation time, further research is desirable to improve the efficiency of computation. The GA is also tested on the DVRP with real-time customer requests and real-time variations in travel times between customers. In this last case the travel times between the nodes are calculated using a time-dependent shortest path algorithm and form the input to the vehicle routing problem algorithm in a matrix form.

More recently Donati *et al.* (2008) use the ACO metaheuristic to work on the TDVRP with hard time windows. New time-dependent iterated local search procedures are introduced to perform efficiently in terms of computation times and quality of the solutions. They take the travel time distribution as a continuous function by deriving the discrete speed distribution. The TDVRP is approximately solved within reasonable computation time for 30 customers using a real life network in Padua (Italy), with speeds collected hourly from the traffic control system Cartesio. To compute the time dependent paths between customers, the model is integrated with a robust shortest path algorithm that stores a set of paths that represent a suitable approximation of the problem. That is, when the departure time is known, the proper pre-calculated path will be selected and the proper travel time distribution calculated to reflect the path follows.

Van Woensel *et al.* (2008) use a TS metaheuristic to handle the TDVRP without time windows. To determine travel speeds, approximations based on queuing theory are used. They show that their queuing approach has significant gains compared to other time-dependent modelling approaches. They mention that in most real-life cases, speeds are not available or are only available for a small part of the road network. Flows, which are much easier to measure and estimate, are however more often available. The queueing approach offers in this case a valuable model for translating flows into speeds. They test their solution method successfully (using a real life network in Belgium) for up to 80 customers with 144 time-zones of 10 minutes. In this experiment they assume that the distance between customers are fixed and have to be known. Their future research path involves applying the developed approach to the DVRP with time windows.

Further research about iterated local search procedure for the TDVRP is done by Hashimoto *et al.* (2008). This research proposes a filtering method to restrict the search space in the neighbourhoods to avoid many solutions having no prospect of improvement. Moreover, they use efficient local improvement heuristics as the 2-opt heuristic and cross exchange heuristic. The iterated local search procedure is first tested for time-independent instances for which the computational results indicates that the proposed method is highly efficient. Subsequently the procedure is used for the TDVRP with 100 customers and 3 time-zones in which the shortest paths over the network are known a priori.

Figliozzi (2009) presents time-dependent instances with 100 customers and solutions using new constructive methods. He uses a route construction method, where at each iteration the customer who adds the minimum amount of cost will be visited next. No iterated local search procedure is used. All moves are on route level, the improvements work on a subset of routes. The idea is to combine several routes with prospect of improvement by taking all customers of these routes together. Subsequently, he creates new routes using the route construction method. The computational results indicate that the proposed algorithms can handle the soft and the hard time window TDVRP in relatively small computation times. In these experiments, distances between customers are fixed, i.e., no shortest path calculations will be done. The analysis and experimental results of the computational complexity indicate that average computation time increases quadratically with the number of customers.

In contrast to the papers cited above, which provide heuristic procedures for the solution of the TDVRP, Soler *et al.* (2009) present a way to optimally solve the TDVRP, at least for small size instances due to its complexity. They deal with the problem by transforming it into a GVRP and then transform the obtained GVRP into an ACVRP, a VRP without time windows restrictions for which several exact procedures have been tested. Thus, they can solve the TDVRP with existing algorithms for the ACVRP (as those cited in Subsection 2-1-2) both heuristically and optimally, in this last case at least for small size instances. Notice that the mentioned paper is a theoretical work, with the aim that its results can be used in the future as ideas or tools to test the efficiency of specific procedures for the TDVRP. Until now, to the best of our knowledge, no computational experiments using the work of Soler *et al.* (2009) are reported on.

2-2-2 Lessons Learned

From the literature review, it is obvious that the usual VRP metaheuristics are not able to deal with time-dependent travel times without essential structural modifications. Local improvement heuristics do not only influence the travel times of a few exchanged customers, but may now have impact on all other customers on the route. Due to the possibility of arrivals in other time-zones, the travel time has become variable. In this way, the number of required travel time calculations and shortest path calculations becomes a critical factor, that has a big influence on the increase of computation time. Nevertheless, Ichoua *et al.* (2003) use local improvement heuristic on the TDVRP in a TS environment. Next, Fleischmann *et al.* (2004) with focus on a local improvement heuristic, Haghani and Jung (2005) using GA, Donati *et al.* (2008) using ACO, Van Woensel *et al.* (2008) using TS, and Hashimoto *et al.* (2008) with focus on local improvement heuristics, follow them. However, none of them calculate the shortest paths on the fly, i.e., at departure time from a location to the next. All of them store one or a set of paths before the start of the VRP calculation.

Malandraki and Daskin (1992) were the first who focused on the TDVRP. They use GRAP without local improvement heuristics and mention that more research needs to be devoted to the development of algorithms for the TDVRP. This recommendation is taken to heart by the earlier mentioned researchers.

Later Figliozzi (2009) proposes a solution method that needs no local improvement heuristics and the solution quality seems to be at least so good as previously reported results. He uses only improvements on route level. Note that also Figliozzi (2009) does not perform a shortest path calculations during the TDVRP process.

Moreover an extra consequence of having time-dependent travel time is that a proposed method does not automatically satisfy the FIFO property. Ichoua *et al.* (2003) were the first who guaranteed the FIFO property and did this in such a smart way that a lot of researchers until Figliozzi (2009) copied their idea. The concept is to take the travel time distribution as a continuous

function by deriving the discrete speed distribution. Only Fleischmann *et al.* (2004) use their own developed smoothed time function.

More lessons that can be learned come from the work of Fleischmann *et al.* (2004). They reduce the number of time-zones by joining time-zones with similar travel times.

Finally Soler *et al.* (2009) had a clever idea to make a transformation from the TDVRP into the GVRP and then into the ACVRP. The last is solvable with familiar methods that can handle problems faster than the direct methods found for the TDVRP. Notice that it is a challenge to use this idea, because the implementation of the results seems non-trivial.

Chapter 3

Methodology RESPONSE™

TNO BU M&L has developed a logistics optimisation model to approximately solve the Vehicle Routing Problem, called RESPONSE™. This chapter presents the methodology used in this model. Section 3-1 gives an introduction of the model. Section 3-2 formulates the methodology used in the model based on the work of Uil (2004). This chapter ends with an discussion in Section 3-3.

3-1 Introduction

RESPONSE™ searches for possibilities to reduce cost in a company specific network. In order to realise this, it must be determined which depot services each customer, how many vehicles are needed and which routes are used. This problem can be decomposed by first assigning each customer to a depot using the RESPONSE™ Facility Location Model (FLM) tool and next using the RESPONSE™ VRP tool for solving the resulting VRP to obtain the minimal number of routes with the minimal driving time. Finally the minimal number of vehicles needed are calculated using the slack in the trips. If the approximate transportation cost of the FLM tool differ considerably from the transportation cost of the VRP tool, a feedback to the FLM tool is done, in which the distribution cost will be updated, until the transportation cost of both tools are similar.

RESPONSE™ is successfully applied to many different real life situations where strategic questions, as well as tactical and operational questions are an issue. An example of a strategic questions is 'How many depots are needed and where should they be located in order to retain a given service level?'. A tactical and operational question could be 'What is the optimal composition of the vehicle fleet?' or 'What is the extra cost of visiting a new customer?'. Van Staalduinen (2009) presents more information about the application of RESPONSE™.

This thesis focuses on the VRP, therefore from now the FLM tool is taken out of consideration. The VRP tool is written in the object-oriented computer programming language Visual Basic .NET (VB.NET) and is based on an ES metaheuristic proposed by Homberger and Gehring (1999). They present two ways to formulate the VRPTW problem. The implementation of RESPONSE™ is based on the ES2 metaheuristic in their article.

RESPONSE™ currently uses a road network based on constant speeds, i.e., the travel times are not time-dependent. To model congestion, the speeds on the network are determined by multiplying the off-peak hours speeds with a congestion factor (between 0 and 1) for specific areas. Notice that during the scheduling horizon the speeds will remain constant. Clearly, this is just a rough approximation of actual conditions. RESPONSE™ can currently approximately

solve a VRP with 100 customers and 15 routes within 10 minutes on an Intel Core 2, 2.4 GHz computer. It seems the results of RESPONSE™ are of reasonable quality when comparing them with other existing methods.

3-2 The Model

The previous section presents a short description of RESPONSE™. This section gives detailed information on the methodology used in RESPONSE™based on the work of Uil (2004). Subsection 3-2-1 formulates the objective and constraints. The problem is separated by first solving a VRP with only one route per vehicle. From this solution, a new solution that possibly contains multiple routes per vehicle can be created. Subsection 3-2-2 takes more interest in the used solution method, i.e., ES, for solving the VRP with only one route per vehicle. Subsection 3-2-3 approximately solves the VRP with multiple routes per vehicle.

3-2-1 Mathematical Formulation

RESPONSE™ is constructed in such a way that it can approximately solve a homogeneous single depot VRPTW with pick-up and delivery using multiple routes per vehicle. This gives the following conditions:

- there is one depot with opening time, closing time and load time per vehicle;
- there are n customers that need to be serviced from the depot;
- each customer has a time window in which it has to be serviced;
- each customer has a specific fixed service time;
- there is an infinite vehicle fleet containing one vehicle type with a fixed capacity;
- each customer has a demand and supply smaller than the capacity of the vehicle;
- a customer order is serviced in one stop;
- fixed distances and travel times between all locations (depot and customers).

First RESPONSE™ works on the problem without multiple routes per vehicle. The mathematical formulation for this problem can be formulated using two types of variables, introduced by Uil (2004). Endogenous variables, that variate during the VRP process and the exogenous variables, that remain constant. The mathematical formulation defined using these variables uses a slightly uncommon style in notation, this is done to present the way in which RESPONSE™ calculates the solution, in detail. Especially notice the used indices.

The used endogenous variables are

N_R	variable indicating the needed number of routes;
$N_C(r)$	variable indicating the number of customers in route r;
$id(r, c)$	variable indicating the ID of the c^{th} customer that is visited in route r;
i^-	the ID (depot or customer) of the predecessor of customer i;
i^+	the ID (depot or customer) of the successor of customer i;
epd_{depot}	earliest possible departure time, i.e., earliest time a vehicle can leave the depot;
epd_i	earliest possible departure time, i.e., earliest time that a vehicle can leave customer i, depending on previous customers in the route and restrictions at the depot;

lfa_{depot}	latest feasible arrival time, i.e., latest time that a vehicle must be back at the depot.
lfa_i	latest feasible arrival time, i.e., latest time a vehicle must arrive at customer i to be able to service customer i and all its successors and be back at the depot on time;
$t_L(r,c)$	time that the vehicle leaves from the c^{th} customer or depot in route r;
l_{depot}	load time at the depot;
s_i	service time for the i^{th} customer.

In addition, the used exogenous variables are

c	customer/depot index (element of N);
r	route index (element of N^+);
i	an ID of a customer (element of I);
$depotID$	the ID of the depot;
cpr	cost per route;
cph	cost per hour;
t_{c_1,c_2}	the time needed to travel from customer or depot c_1 to customer or depot c_2;
$demand_c$	the demand of the customer with ID c;
$supply_c$	the supply of the customer with ID c;
$capacity$	the capacity of the vehicle;
LB_{depot}	lower bound, i.e., opening time of the depot;
UB_{depot}	upper bound, i.e., closing time of the depot;
LB_i	lower bound, i.e., start of the service time window of customer i;
UB_i	upper bound, i.e., end of the service time window of customer i;
N	set of natural numbers $= \{0, 1, 2, \cdots\}$;
N^+	set of positive natural numbers $= \{1, 2, \cdots\}$;
I	set of customer IDs.

The mathematical formulation becomes

minimise:

$$cpr \cdot N_R + cph \cdot \sum_{r=1}^{N_R} \left(t_L\left(r, N_C(r)\right) + t_{id(r,N_C(r)),depotID} - t_L(r,0) \right), \tag{3-1}$$

subject to:

$$\sum_{c=1}^{N_C(r)} demand_{id(r,c)} \leq capacity \qquad \forall r \mid 1 \leq r \leq N_R \tag{3-2}$$

$$\sum_{c=1}^{i} supply_{id(r,c)} + \sum_{c=i+1}^{N_C(r)} demand_{id(r,c)} \leq capacity \qquad \forall r, i \;\Big|\; \begin{array}{l} 1 \leq r \leq N_R \\ 1 \leq i \leq N_C(r) - 1 \end{array} \tag{3-3}$$

$$\sum_{c=1}^{N_C(r)} supply_{id(r,c)} \leq capacity \qquad \forall r \mid 1 \leq r \leq N_R \tag{3-4}$$

$$\sum_{r=1}^{N_R} \sum_{c=1}^{N_C(r)} V(r,c,i) = 1, \text{ where } V(r,c,i) = \begin{cases} 1 & \text{if } id(r,c) = i \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in I \tag{3-5}$$

$$id(r,0) = depotID \qquad \forall r \mid 1 \leq r \leq N_R \tag{3-6}$$

$$id(r, N_C(r)+1) = depotID \qquad \forall r \mid 1 \leq r \leq N_R \tag{3-7}$$

$$i^- = id(r,c) \left| \begin{array}{l} id(r,c+1) = i \\ r \leq N_R \\ 0 \leq c \leq N_C(r) \end{array} \right. \quad \forall i \in I \tag{3-8}$$

$$i^+ = id(r,c) \left| \begin{array}{l} id(r,c-1) = i \\ r \leq N_R \\ 1 \leq c \leq N_C(r)+1 \end{array} \right. \quad \forall i \in I \tag{3-9}$$

$$epd_{depot} = LB_{depot} + l_{depot} \tag{3-10}$$

$$epd_i = \max(epd_{i^-} + t_{i^-,i}, LB_i) + s_i \quad \forall i \in I \tag{3-11}$$

$$lfa_{depot} = UB_{depot} \tag{3-12}$$

$$lfa_i = \min(lfa_{i^+} - t_{i,i^+} - s_i, UB_i) \quad \forall i \in I \tag{3-13}$$

$$epd_i \leq lfa_i + s_i \quad \forall i \in I \tag{3-14}$$

$$t_L(r,0) = lfa_{id(r,1)} - t_{id(r,0),id(r,1)} \quad \forall r \mid 1 \leq r \leq N_R \tag{3-15}$$

$$t_L(r,c) = \max\left(t_L(r,c-1) + t_{id(r,c-1),id(r,c)}, LB_{id(r,c)}\right) + \ldots + s_{id(r,c)} \quad \forall r,c \left| \begin{array}{l} 1 \leq r \leq N_R \\ 1 \leq c \leq N_C(r) \end{array} \right. \tag{3-16}$$

$$N_R \in N^+ \tag{3-17}$$

$$N_C(r) \in N^+ \quad \forall r \mid 1 \leq r \leq N_R \tag{3-18}$$

$$id(r,c) \in I \quad \forall r,c \left| \begin{array}{l} 1 \leq r \leq N_R \\ 1 \leq c \leq N_C(r). \end{array} \right. \tag{3-19}$$

The objective of the model (3-1) is first to minimise the number of routes due to the high cost per route (cpr) and than to minimise the total travel time over all routes.

The first three constraints make sure that the amount of products in the vehicle never exceeds the vehicle capacity. Constraint 3-2 states that the total demand of the customers in one route may never exceeds the vehicle capacity. This total demand equals, due to the single sourcing of the pick-up and delivery, the amount of loaded products by departure from the depot. Constraint 3-3 also takes into account the supplied products from the customers. Because of the supplied products, after each customer departure a feasibility check for the capacity is needed. Constraints 3-4 check the last customer departure to the depot for each route. In this last travel path, all delivery products are already serviced, the vehicle contains now only all pick-up products.

Constraint 3-5 requires that every customer has to be visited exactly once by exactly one vehicle. This constraint also forces that the number of customer IDs equals the total number of customers serviced over all routes. Constraints 3-6 and 3-7 force each route to start and end at the depot. Constraints 3-8 and 3-9 define the succeeding and preceding customer to each customer.

The model does not have to imply start and end times for each route, because typically a route will hold some slack and therefore many different departure times will result in the same total travel time. The variables epd and lfa are used to check if a solution is feasible, i.e., the routes may not violate lower and upper bounds of the customers time window and the depot time window. The variable epd is set per customer by constraints 3-10 and 3-11 and give the earliest possible departure time that the allocated vehicle can leave the customer, considering the route to the customer. The variable lfa is set per customer by constraints 3-12 and 3-13 and gives the latest feasible arrival time that the vehicle must arrive at the customer to still be able to make the rest of the route in time. A route is feasible when the route from the begin depot to each customer does not violate the route from the customer to the end depot. This is the case when the earliest possible departure time is not greater then the latest feasible departure time,

this is implied by constraint 3-14. Notice that the latest feasible departure time equals latest feasible arrival time plus the service time.

As mentioned above, a route does not have to result in unique departure and arrival times at the customers, but a travel schedule of the route to calculate its total travel time is needed. Therefore a variable leave time is introduced per stop in each route, that gives a possible time to leave the current stop in a route and results in the minimal travel time. Constraints 3-15 and 3-16 set these variables.

Finally constraints 3-17, 3-18 and 3-19 set the explanatory variables to correct target values.

3-2-2 Evolution Strategy

As mentioned in Section 3-1, RESPONSE^{TM} is based on an ES metaheuristic, i.e., the ES2 proposed by Homberger and Gehring (1999). The global idea of ES is presented in Subsection 2-1-2 with subsection subtitle 'Evolutionary Algorithm'. Recall that ES contains four processes: generating a start population, recombination, mutation and selection respectively. In this subsection, these processes are applied to the VRP with only one route per vehicle, formulated in the previous subsection. Before the processes of ES can be started, first a definition of an individual and a population has to be made. The way in which RESPONSE^{TM} performs this definition and the 4 processes, inspired by the work of Homberger and Gehring (1999) is written below.

Definition of Population and Individual

A set of solutions to the VRP is called a population and a single solution is called an individual. This individual consists of a set of routes and can be represented as

$$\begin{array}{ll} N_R & \\ N_C(r) & \forall r \mid 1 \leq r \leq N_R \\ id(r,c) & \forall r,c \left| \begin{array}{l} 1 \leq r \leq N_R \\ 1 \leq c \leq N_C(r). \end{array} \right. \end{array} \tag{3-20}$$

This makes it possible to represent the population as

$$\begin{array}{ll} N_{Ind} & \\ N_R(i) & \forall i \mid 1 \leq i \leq N_{Ind} \\ N_C(i,r) & \forall i,r \left| \begin{array}{l} 1 \leq i \leq N_{Ind} \\ 1 \leq r \leq N_R(i) \end{array} \right. \\ id(i,r,c) & \forall i,r,c \left| \begin{array}{l} 1 \leq i \leq N_{Ind} \\ 1 \leq r \leq N_R(i) \\ 1 \leq c \leq N_C(i,r). \end{array} \right. \end{array} \tag{3-21}$$

In this formulation, the variable N_{Ind} indicates the number of individuals. The possible routes with customers are now individual dependent. It is not shown that the possible routes represent feasible solutions to the VRP. This is not a problem, because in the generation of the routes feasibility in each step is maintained.

In the recombination process (process 2), the new offspring solution is not directly recombined from the parent population. Even if this is the case, this mostly results in infeasible solutions. Therefore, for each solution two extra properties are given. The first is a mutation code and the second is the route elimination parameter. The mutation code is an array that contains all customers IDs twice in a seemingly random order and is used in the mutation phase to restructure a solution. The route elimination parameter can have the value 0 or 1, which indicates if the mutation phase should end with an attempt to reduce the number of routes. These two properties are added to the previous definition of individual and population. The formulation of the properties is given by

$$\begin{array}{ll} RouteEliminaton(i) & \forall i \mid 1 \leq i \leq N_{Ind} \\ MutationCode(i,c) & \forall i,c \left| \begin{array}{l} 1 \leq i \leq N_{Ind} \\ 1 \leq c \leq 2 \cdot Cardinality_I. \end{array} \right. \end{array} \tag{3-22}$$

Here, $Cardinality_I$ means the total number of customers in the collection I.

Process 1: Generate Starting Population

A starting population is a set of different feasible routes. The size of the population can be chosen arbitrarily, but has to be big enough to ensure enough diversity and a significant solution space that can be searched. A too big population is also not a good choice, because it will cause slow convergence towards good solutions.

A start solution is generated in the following 3 steps:

1. shuffle all customers in random order and insert them to create the routes;
2. create the mutation code by shuffling the set of two times all customers in random order;
3. randomly select 0 or 1 to create the route elimination parameter.

The insertion procedure in step 1 does need extra explanation. For the insertion of a customer, all possible positions a customer can be inserted at need to be investigated, i.e., for each position in each route, the following conditions need to be checked:

- insertion of a customer never violates capacity constraints 3-2, 3-3 and 3-4;
- insertion of a customer never violates time window constraints 3-14.

If both conditions are met, the customer can be inserted at the given position. If there is no position that the customer can be inserted at, a new route is created from the depot to the new customer and back. If there is only one position that meets both conditions, then the customer is inserted at that position. Mostly there are several options to insert the customer, in which case the 'greedy' best option that minimises the extra travel time and distance caused by the insertion will be chosen. The calculation of these extra travel time and distance is given below.

The extra distance of an insertion can easily and efficiently be calculated. If customer i is inserted between customers j and k in a route, then the extra distance equals

$$ExtraDistance = distance_{j,i} + distance_{i,k} - distance_{j,k}. \tag{3-23}$$

Unfortunately it is a bit more difficult to calculate the extra travel time of the insertion of the customer. The travel time of a route is not always a direct result of the visited customers

location, it also depends on the time the route starts. To calculate the minimal travel time, the following procedure will be done. First, the latest feasible arrival time of the first customer (calculated by constraints 3-12 and 3-13) is taken as the time the vehicle arrives at this customer. Next, from this moment an extrapolation can be done to the other leave times and arrival times of the route. The leave time from each location (depot or customer) can be calculated using constraints 3-15 and 3-16. By the definition of lfa, this procedure determined the minimal travel time. This procedure is rather unfortunate, because many steps are needed for the calculation of the travel time, which results in poor performance.

Fortunately, next to explicitly calculating the travel time, there is another way to say something about the change in travel time. The change in total travel time is always initiated by the shift in the lfa of the predecessor of the inserted customer as well as the shift in the *epd* of the successor of the inserted customer. These variables are defined in equations 3-24 and 3-25.

$$lfa\text{Shift}_{r,i,c} = lfa_{r,i-1} \; - lfa_{r',i-1} \tag{3-24}$$

$$epd\text{Shift}_{r,i,c} = epd_{r',i+1} - epd_{r,i+1}. \tag{3-25}$$

Here, $lfa\text{Shift}_{r,i,c}$ and $epd\text{Shift}_{r,i,c}$ stand for the shift in lfa respectively *epd* when inserting customer c in route r at position i. Moreover, r' stands for the altered route after the insertion. The greater the change in one of these variables, the greater the increase of the travel time. Because the lfa shift and *epd* shift can be explored in only one step, the use of this variables keep a good performance.

As a consequence, the 'greedy' best position to insert the customer will be determined by the following selection procedure. A customer will be inserted at a certain route position if the lfa shift plus *epd* shift is minimal compared to all other feasible insert positions. If the change in lfa shift plus *epd* shift is minimal at more than one position, the position with the least increase in travel distance will be chosen. If this final criterion has also not left one insert possibility, the position will be randomly chosen from the remaining best options.

Process 2: Recombination

The recombination process consists of two steps. The first is the actual recombination that combines the mutation code from two parents to a new one for a new individual. The second step adds routes and the route elimination parameter from a third parent. The process is shown in Figure 3-1.

For the recombination of the mutation codes, a uniform order based crossover algorithm will be used. This is a crossover operation between the two parent codes, where the ordering of the elements is pretty much preserved.

The crossover operation first creates a mask array, containing randomly ordered zeros and ones, of the same length as the mutation code arrays of the two parents. Next, the elements of the first parent that are in the position where the mask array contains ones are copied to the child array. Finally the empty spots in the child array are filled by the missing elements in the order of the second parent. This operation is illustrated in Figure 3-2. Note that, when copying the missing elements from the second parent, the second element of parent 2 ("1") is already twice present in the child, therefore the second element of the child becomes the next element of parent 2 ("2" instead of "1").

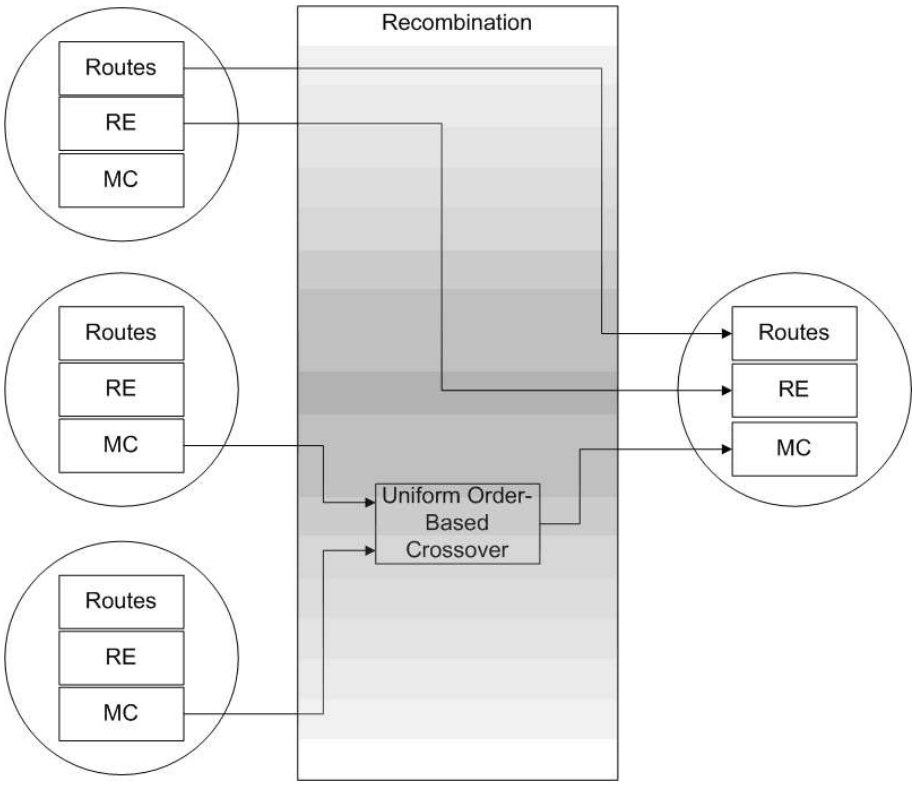

Figure 3-1: The general structure of an evolution strategy metaheuristic.

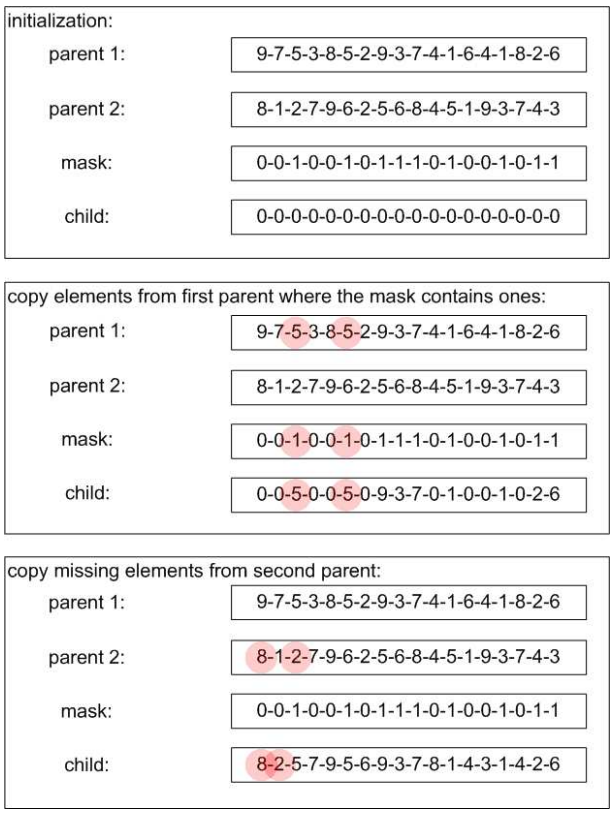

Figure 3-2: The three steps in the uniform order based crossover algorithm. In the second and third step the first two copy actions are indicated.

Process 3: Mutation

The mutation process changes every individual from the offspring population (constructed by the recombination process). An individual is mutated by removal and insertion of every customer in the order given by the mutation code. As described before, the mutation code is an array that lists every customer twice. The procedure loops over the mutation code and performs an action on the customer at the current position. If it is the first time that the customer is encountered, the customer is deleted from its current route and if it is the second time the customer is inserted back into solution. The same insertion procedure is used as is described previously in this subsection (3-2-2) with paragraph title 'Generate Starting Population'.

For example, if the mutation code of a solution is equal to the mutation code of the created child in Figure 3-2 (mutation code = 8-2-5-7-9-5-6-9-3-7-8-1-4-3-1-4-2-6), then the following actions would be performed on the set of routes:

1. remove customers 8, 2, 5, 7 and 9;
2. insert customer 5;
3. remove customer 6;
4. insert customer 9;
5. remove customer 3;
6. insert customer 7 and 8;
7. remove customer 1 and 4;
8. insert customer 3,1,4,2 and 6.

When for each individual the mutation using the mutation code is finished, an attempt is made to eliminate a route from each solution if the *RouteElimination* parameter of the individual is equal to 1. The elimination attempt takes the route with the smallest number of stops and tries to insert the customers in the other routes. If all customers can be fit in the other routes, the reduced solution will replace the solution before the elimination attempt in the offspring population. If it is not possible to remove a route, the offspring population remains the same.

Process 4: Selection

In the selection process all solutions in the offspring population set are rated on the following criteria:

1. minimal total trip times.
2. minimal number of routes;
3. minimal delay factor;

To guide the search of the algorithm to a minimal number of routes, next to criterion 2, criterion 3 is used. The delay factor gives an interpretation of the possibility that a route can be removed from the solution. The delay of a solution is defined as:

$$SolutionDelay(i) = \sum_{c \in S} Delay(i, c). \tag{3-26}$$

Here, S is the set of all customers from the smallest trip of solution i. Then, $Delay(i, c)$ is the smallest delay of inserting customer c in another route of solution i than its own. Its value is equal to 0, if customer c can be inserted in at least one other route of solution i, without violating any time window constraints. A $Delay(i, c)$ larger than 0 equals the smallest time window constraint violation cost by the customer c being inserted in another route than its own. As follows from definition 3-26, the delay of a solution is the sum of delays of all $c \in S$. However, if all capacity in the smallest trip is used, the opportunity that a route can possibly be removed is very small. In this last case, the value of the solution delay is set to ∞.

The selection process generates a new parent population in two steps. For this, two evolution strategy parameters, μ and κ are needed. Here, $\mu + \kappa$ equals the size of the new parent population. In the first step, μ solutions from the offspring population that perform best on criterion 2 will be added to the new parent population. If this criterion gives no unique selection, criterion 1 is applied. The second step completes the new parent population by adding μ solutions. This selection is also based on criterion 2. But now, if this criterion gives no unique selection, criterion 3 is applied. The selection process always orders the new parent population based on 1, with the best known solution at the top.

When the new parent population is created, one can try to improve the quality of the best solution by repeating processes two to four on the new parent population. Each repeat is called an iteration. The number of iterations can be chosen arbitrarily just like the size of the starting population needed in the first process.

3-2-3 Multiple Routes per Vehicle

The previous subsection has shown in which way RESPONSE™ creates a solution for the VRP without multiple routes per vehicle. The obtained solution contains optimal routes, which do not have a fixed begin or end time. Each route actually can be started anywhere between the earliest possible departure time (epd_{depot}) and the latest possible departure time (derived from lfa_{depot}). But the minimal travel time is only reached when the route starts anywhere between the earliest minimal departure time and the latest minimal departure time. The latter is given by $t_L(r, 0)$, which is previously defined by equation 3-15. The former ($t_L^{early}(r, 0)$) can be calculated by

$$t_L^{early}(r, N_C(r)) = epd_{id(r,N_C(r))} \qquad \forall r \mid 1 \leq r \leq N_R \tag{3-27}$$

$$t_L^{early}(r, c) = \min\Big(t_L^{early}(r, c+1) - s_{id(r,c+1)} + \ldots - t_{id(r,c),id(r,c+1)}, UB_{id(r,c)} + s_{id(r,c)}\Big) \qquad \forall r, c \;\Big|\; \begin{array}{l} 1 \leq r \leq N_R \\ 0 \leq c \leq N_C(r) - 1. \end{array} \tag{3-28}$$

Each route constructed by the solution method of the previous subsection is associated with only one vehicle. In order to handle the problem with multiple routes per vehicle, a scenario has to be found where as many routes as possible can be serviced by one vehicle, i.e., a route to vehicle allocation problem occurs. This can be done by looking at each route as a task that needs to start between $t_L^{early}(r, 0)$ and $t_L(r, 0)$, where the length of the task is equal to the total travel time. With this in mind, the minimal number of vehicles have to be found by looking for the minimal number of vehicles that are needed to service these tasks. The intermediate times between the tasks, must be equal to the load time. As a result, the route to vehicle allocation problem can be formulated as a VRP just like handled in the previous subsection, in which the following assumptions are met:

- the routes are modelled as the new 'customers';

- the $t_L^{early}(r,0)$ is the lower bound of 'customer' r, i.e., LB_r;
- the $t_L(r,0)$ is the upper bound of 'customer' r, i.e., UB_r;
- the load time at the depot is the travel time between all 'customers', i.e., $t_{r_1,r_2} \forall r_1, r_2 \in I$;
- the total travel time of route r is the service time of 'customer' r, i.e., s_r.

After reformulation, the problem can be approximately solved using the same ES metaheuristics as in the previous subsection. The resulting variables are translated to find the required number of vehicles and the final route schedule. This translation is listed in Table 3-3. Note that using the same VRP techniques to solve the route to vehicle allocation problem leads to a solution with multiple routes per vehicle.

Table 3-3: Translation of the resulting variables of the 'VRP formulated in the previous section' to the 'route to vehicle allocation problem'.

Resulting variable	Translation
N_R	the needed number of vehicles
$N_C(r)$	the number of routes for each vehicle
$id(r,c)$	the c^{th} route of vehicle r
$t_L^{early}(r,c)$	the start time of the c^{th} route of vehicle r

3-3 Discussion

In this chapter the way in which Uil (2004) created RESPONSE^{TM} to solve the VRP is described. This gives insight into the complexity of solving a VRP and I use it to devise a solution algorithm for a TDVRP. In the next chapter this solution method is described, implemented and tested.

Chapter 4

Time-Dependency related to a Small Problem

This chapter describes the solution method, developed during this masters thesis, for vehicle routing problems with time-dependent travel times tested on a small problem with 100 customers. Section 4-1 reports on the choice and implementation of the solution method. Section 4-2 formulates the small problem. This chapter ends with the computational results obtained using the solution method on the small problem in Section 4-3.

4-1 Solution Method Trade-Off

The previous chapter has described the methodology used in RESPONSE™. From this, it is clear that the current version of RESPONSE™ is not able to consider time-dependent travel times without essential structural modifications. For instance, local changes of a route, as insertions or removals of customers, do not only influence the travel times around the inserted or removed customers, but may now have impact on all other customers on the route. This section describes the choice and implementation of the solution method for the TDVRP obtained from the lessons learned in Subsection 2-2-2. First, Subsection 4-1-1 gives some remarks on the shortest path calculation that should be adapted when using time-dependent travel times. Next, Subsection 4-1-2 searches for an efficient solution method for the TDVRP in which the time-dependent shortest path calculation can be executed. Finally, Subsection 4-1-3 presents a successful solution method based on the work of Figliozzi (2009).

4-1-1 Time-Dependent Shortest Path Calculation

From the last condition given in Subsection 3-2-1 it can be concluded that fixed distances and travel times between all locations must be known in order to approximately solve the VRP from Chapter 3. These distances and travel times are stored in, respectively, a distance matrix and a travel time matrix. In some cases the fixed distances and travel times are given in advance (simple experiments), but in other cases only the distances and speeds over the road network are given (real-life experiments). In the latter, an arc between two customers can possibly be constructed by several links over the network. The combination of links to create the shortest path (in time), can be calculated using a shortest path algorithm. This shortest path algorithm is included in RESPONSE™; it is the implementation of the work of Dijkstra (1959). Due to

the fixed speeds over the network, the distance and travel time matrices have to be calculated once before the start of the VRP process.

To handle the TDVRP in real-life experiments, the travel time is dependent on the departure time, as well as the time dependent speed. The time dependent speed has influence on the shortest path and hence it also becomes time dependent. From Subsection 2-2-2 can be concluded that none of the researchers do shortest path calculations during the TDVRP process. Most of them store one path before the start of the TDVRP calculation and only some (like Donati *et al.* (2008)) store a set of paths that represent a suitable approximation of the problem, such that the proper pre-calculated path from the set of paths will be selected given the departure time. However when the departure time is such that a time-zone border will be crossed, the pre-calculated shortest path is possibly not the shortest any more. This effect becomes even larger for real-life problems, since congestions cause large speed variations over the planning horizon and solutions using the pre-calculated shortest paths may become inefficient. The goal of this thesis is to obtain good quality solutions to real-life problems. This is why the choice is made to calculate the shortest paths during the TDVRP process, i.e., at departure time from a location to the next. Therefore, a shortest path function depending also on the departure time is created that can be applied in the TDVRP tool. This implies that during the TDVRP process the time dependent shortest path algorithm will be called a lot of times. This has impact on the calculation time, however the increase in efficiency of the solution is chosen to be more important in this thesis. The increase in computation time is probably the reason why other researchers refrain from this approach.

The time dependent shortest path algorithm calculates for a specific departure time the shortest path between two customers. This path and each of its links, also needs to satisfy the FIFO property. This can be done using the idea of Ichoua *et al.* (2003) formulated in Section 2-2. However, he uses it for a given shortest path and for this thesis it is integrated into the shortest path algorithm. To clarify the procedure to satisfy the FIFO property, some pseudo-code will follow. The procedure describes how to calculate a time-dependent travel time and the function is a recursive function that is applied in the procedure to satisfy the FIFO property. The actual VB code is included in appendices A-1 and A-2.

```
procedure Calculation of the time-dependent travel time
T = time-zone at departure
    if arrival is in T
        travel time = length_link / speed_link(T)
    else
        travel time = function Time-zone crossing
    end if
end

function Time-zone crossing
travel time until time-zone border (t1) = subtract departure time from border time
distance until time-zone border (x1) = t1* speed_link(T)
remaining travel time = (length_link - x1) / speed_link(T+)
    if arrival is in T+
        travel time = length_link / speed_link(T+)
    else
        travel time = function Time-zone crossing
    end if
end
```

4-1-2 Search for an Efficient Solution Method

The previous subsection shows how to calculate the time-dependent shortest path and travel times. Now, a solution method is needed in which these paths and travel times can be integrated. Here, local improvement heuristics have to be avoided due to the big influence on computation time mentioned in Subsection 2-2-2. From this point of view, the approach of Soler *et al.* (2009) seems very promising, but first more research on how to implement this work has to be done.

As previously mentioned, the implementation of the work of Soler *et al.* (2009) seems non-trivial. At a first glance, the difficulty seems to lie in the transformation. However, further research turns out that the difficulty lays not in the transformation but in handling the resulting ACVRP.

The transformation from TDVRP to GVRP can be visually interpreted as transforming one TDVRP node into several GVRP nodes. Each created node represents a different time-zone at the same location of the original TDVRP node. The new nodes derived from one TDVRP node become a cluster. In this way, multiple clusters can be created and the GVRP is defined.

During the transformation from GVRP to ACVRP the clusters remain the same. But where in the GVRP exactly one node from each cluster has to be visit, in the ACVRP all nodes over all clusters have to be visited. The work of Soler *et al.* (2009) illustrates an ACVRP solution of a small theoretical example, that is presented in Figure 4-1.

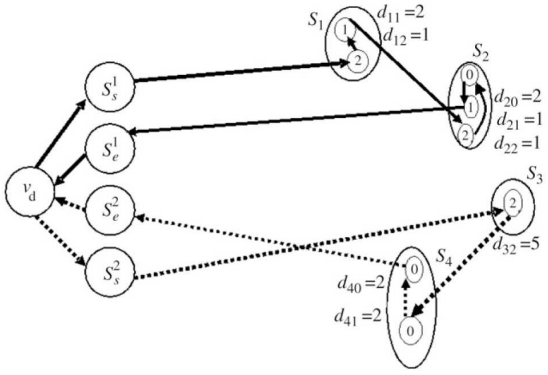

Figure 4-1: An optimal ACVRP solution constructed by the transformation formulated by Soler *et al.* (2009).

In Figure 4-1, v_d represents the depot vertex. S_s^1 and S_s^2 indicate the start (departure time from the depot) of vehicle 1 resp. 2. S_e^1 and S_e^2 indicate the end (arrival time at the depot) of vehicle 1 resp. 2. There are 4 clusters, clusters S_1 to S_4 and each cluster contains several nodes. The example has integer values for the travel times and the time windows for practical reasons. Every node corresponds to one time-unit and the access information from node to node is saved before the start of the TDVRP tool. The problem is disconnected from time windows restrictions. This is needed, because inside a cluster it must be allowed to go from a later time-unit to an earlier one. If a vehicle enters a new cluster, all nodes in this cluster have to be consecutively visited before leaving this cluster. The solid line in the figure represent the route of vehicle 1 and the dotted line represent the route of vehicle 2. Notice that the demand of a customer is split up over all nodes of the specific cluster, e.g., the demand of customer 1 equals $d_{11} + d_{12}$. Moreover, the service time of a customer is included in the travel time from this customer to a next customer, only when travelling to a new cluster. When translating the solution of this ACVRP to the solution of the TDVRP, only the first visited customers of a cluster are kept to construct the routes.

Generally, the ACVRP is a simplification of the VRP variant that can be approximately solved by RESPONSE^{TM}. From this, one could hope that the created ACVRP can be solved with

RESPONSE™. But this optimisation model cannot handle this clustered ACVRP in which it is allowed to go from a later time-unit to an earlier one. For this small theoretical example the created ACVRP could be easily solved as is done in the article of Soler *et al.* (2009). However, to solve a real-life problem, which is in the objective of this thesis, the nodes of the clustered ACVRP include not only one time-unit, but a whole time-zone. In this case it is also important to know in which part of the time-zone the vehicle is leaving, due to satisfaction of the FIFO-property. Therefore the time-zones need to be split up in a lot of time-units, however this requires a lot of memory, and seems difficult to realise in practise.

In conclusion, a clustered ACVRP differs from the general ACVRP. A clustered ACVRP has to save the access information from node to node disconnected from time window restrictions, which cannot be solved with de current RESPONSE™ model. In this way, the transformation does not lead to a VRP that can be solved within our framework. In other words, the theoretical work of Soler *et al.* (2009) is not readily applicable in our case.

Because the application of the theoretical work of Soler *et al.* (2009) is not able to deal with a real-life problem, this work is of less importance for this thesis. The work of Figliozzi (2009) is used instead. Figliozzi (2009) takes a route improvement approach instead of the local improvement approach. This work can be applied successfully, and more information about the implementation is presented in the following subsection.

4-1-3 Successful Solution Method

This subsection describes a successful solution method based on the work of Figliozzi (2009). The difference between this solution method and the solution method of the current RESPONSE™ without time-dependent travel times will be described. First a short description of the methodology used by Figliozzi (2009) is given and then my way of implementing a successful solution method based on that methodology is presented.

Methodology Figliozzi (2009)

As mentioned in Section 2-2, the idea of Figliozzi (2009) is to use a solution method that needs no local improvement heuristics. First, he creates a start solution based on a route construction algorithm. After that, on a subset of routes, improvements are done on route level.

The route construction algorithm starts with a random customer. Next, there is a search for the best order to insert at the next position in the route. The best order is the one with the smallest insertion cost. This insertion cost is dependent on several factors, including the shift in departure time of the last customer, the extra distance travelled from the previous to the new customer, the urgency of serving the customer (the time remaining until the vehicle's last possible service time starts) and the capacity slack of the vehicle after serving the customer. If there is no feasible order found, a new route with a random order that is not yet visited, is created.

When the start solution is created, a route improvement algorithm will be applied. The motivation of Figliozzi (2009) for this algorithm is to combine several routes with prospect of improvement. First, he chooses to take the route with the smallest number of orders. Next, one or more routes nearest to this first smallest route are taken, assuming that the nearest routes have the potential to be improved. Here, the nearest route is determined by calculating the geographical distance between the absolute centre of the routes. When the subset of routes to improve are determined, all customers of these routes will be taken together. On this subset of routes, the route construction algorithm can be applied. Notice that the route improvement algorithm can be applied several times, each time on another subset of routes.

Implementation of the method

Because local improvements will be avoided, the ES algorithm from RESPONSE™ cannot be used. To handle time-dependent travel times, I had to completely restructure RESPONSE™. The new method will be referred to as RESPONSE™ CRIA (Construction and Route Improvement Algorithm) and the current RESPONSE™ model will be referred as RESPONSE™ ES (Evolution Strategy). In this paragraph, the changes from RESPONSE™ ES to RESPONSE™ CRIA will be formulated.

In the first place, the start solution in RESPONSE™ CRIA has to be constructed in a different way than in RESPONSE™ ES. In the latter, as can be read in Subsection 3-2-2, a random order is given and is inserted at the best position. But in RESPONSE™ CRIA, the position will be given (always the last position of a route) and the best order will be inserted. This is recommend, because in the time-dependent case, a customer insertion at another position than the last one of a route, has impact on all succeeding arcs in the route, in the same way local improvement heuristics do. To implement a construction algorithm that contains the above methodology, the procedure Sub FindBestOrderToInsert is written in VB. The code can be found in Appendix A-3. For clarity the construction algorithm with the above methodology is given is pseudo code.

```
procedure Construction of a solution
    for each order selected as starting point in route list
        for each parameters in insertion strategy parameters
            for order index = 1 to number of orders
                function Find the best order to insert at the next position
            next order index
        next parameters
    next order selected as starting point
    if check for improvement of the solution
        use this solution when it is better then the previous best solution
    end if
end

function Find the best order to insert at the next position
    if there is already a trip created with at least one order
        for each order in order list
            if order insertion is feasible
                Calculate insertion cost depending on the parameters
                Remember the order with the least insertion cost
            end if
        next order
    end if
    if there is no feasible order found (or a new trip is needed)
        for each order in order list
            if order insertion is feasible
                Calculate insertion cost depending on the parameters
                Remember the order with the least insertion cost
            end if
        next order
    end if
end
```

The second difference between RESPONSE™ ES and RESPONSE™ CRIA is that, due to the time-dependent travel times, the time shifts on *epd* end lfa from RESPONSE™ ES could no longer be used. This is why in RESPONSE™ CRIA, these time shifts are removed and all routes now have fixed departure times. To construct the routes, a lot of information will be stored for each order to insert, such as the previous leg travel time, the previous leg distance, the arrival time, the waiting time, the start service time, the leg travel time to depot, the leg distance to depot and the departure time. To make the routes as large as possible, each route has to be started at the earliest possible departure time, such that there is no waiting time to service the first customer. For this, a backwards shortest path algorithm is used to calculate this departure time. Because all created routes from the construction algorithm now have fixed departure times, the construction from RESPONSE™ ES that determines multiple routes per vehicle cannot be used any more. However, it is theoretically possible to realise multiple routes per vehicle, when the created routes are full because of capacity constraints. RESPONSE™ CRIA cannot (yet) solve a TDVRP with multiple routes per vehicle.

The third difference between RESPONSE™ ES and RESPONSE™ CRIA is the insertion strategy. In RESPONSE™ ES, the best insertion is the one where the lfa shift plus *epd* shift is minimal. Now, in RESPONSE™ CRIA the best insertion is the one with the smallest insertion cost, if there are multiple best orders, the one with the minimal time-shift is taken. The insertion cost is dependent on several factors based on the methodology used by Figliozzi (2009). The chosen factors are time-shift, extra distance, time-shift from depot to depot, extra distance from depot to depot, delay, remaining capacity and waiting time, which is calculated in Appendix A-3. Here, Sub Overrides CompareInsertPosition (described in A-4) will be executed, in which the insertion cost will be calculated. The relation between the mentioned factors dependent on the search space parameters $\{\delta_0, \delta_1, \delta_2, \delta_3, \delta_4, \delta_5, \delta_6\}$, which are formulated in Appendix A-5. Notice that search space parameters $\{1, 0, 0, 0, 0, 0, 0\}$ lead to the order with the minimal time-shift, also known as the least time insertion strategy. In the same way, search space parameters $\{0, 1, 0, 0, 0, 0, 0\}$ lead to the least distance insertion strategy. The least time insertion strategy and the least distance insertion strategy are based on the time resp. distance travelled from the previous to the new customer. To get the insertion in a different direction, other criteria are chosen for the time resp. distance, namely the time resp. distance travelled from depot to depot, that is assigned to δ_2 resp. δ_3. In some search space parameters, δ_4 is chosen to influence the insertion in such a way that customers have priority when they almost close. Moreover δ_5 is taken to deliver first the customer with as much as possible demand and δ_6 is inserted to avoid needless waiting time. A lot of trail and error has lead to these parameters.

To create a good start solution, it is useful to run the construction algorithm several times, each time starting with another order with prospect of no waiting time (instead of starting with a random order), and each time with a different insertion strategy (instead of using 1 parameter set from the search space).

As follows from the name of RESPONSE™ CRIA, it is a construction and route improvement algorithm. The way to implement the construction algorithm is mentioned above. Next, more information about the route improvement algorithm is given. Again this is illustrated with pseudo code and the VB code corresponding to this route improvement algorithm can be found in Appendix A-6 and A-7.

```
procedure Improvement of a solution
    for each route in smallest, second smallest, largest waiting time
      search for routes near this one
      make subsets containing all orders from selected routes
      for subset = 1 to end
        procedure Construction of a solution
      next subset
      remember the best solution of all subsets
    next route
    for each route in route list
      procedure Construction of a solution (for the orders in only this route)
    next route
    check if the solution has improved
end
```

The improvement algorithm is based on a variation of the idea of Figliozzi (2009). Several subsets of routes from the constructed solution will be taken, and on each subset the route construction algorithm will be applied. Recall that the nearest route is determined by calculating the geographical distance between the absolute centre of the routes and the smallest route has the least number of orders. Take n, the number of routes of the constructed solution, then the first subset contains the route with the smallest number of orders and $n-2$ routes nearest to this first smallest route. In this way, a subset P of $n-1$ routes is created. All next subsets also contains the smallest route, but for each next subset, the furthest route to the first will be removed, until 2 routes are over. These first $n-2$ created subsets, will be referred to as standard subsets. Next, $n-3$ extended subsets will be created containing the same smallest route, but in all subsets leaving out the nearest route to the first. For example, a constructed solution with 5 routes, where route 1 is the smallest, and route 2, 3, 4 and 5 consecutively are the nearest to route 1. Then, the standard subsets become $(1,2,3,4)$, $(1,2,3)$ and $(1,2)$ and the extended subsets become $(1,3,4)$ and $(1,3)$. Next, on each subset, the obtained routes will be reconstructed with the use of the construction algorithm. The best solution will be remained. Next, this best solution is taken as the new constructed best solution and a new route improvement can be done. Now, not taking the smallest route as the first one, but the second smallest one, could also yield potential improvement. For the same reason, the route with most waiting time is taken as first, but only if this route is not the same as the smallest or the second smallest. All of the above improvements will be repeated until there is no improvement any more. Last but not least I try to improve all single routes, by ordering them in another way.

Notice that a lot of improvements are done here. For very small problems up to 25 customers this is no problem. But for problems with 100 customers within 15 routes, the computation time will become problematic. Naturally, the solution has a bigger prospect of improvement when more improvements are done. In the example above, taking all other possible subsets like $(1,4)$ or $(1,3,4,5)$ might lead to a better solution. Due to the increase in computation time, in RESPONSE™ CRIA I only take the most promising subsets to improve the solution.

Moreover, to generate a broader spectrum of solutions, in the insertion strategy I tried to take in some cases the second best order, instead of the best one. For example, on each subset the improvement algorithm can be applied a second time, now chosing the second best order as the first order in the second route. Because the poor quality of solutions in relation with the large computation time, these kind of improvements will not be done in RESPONSE™ CRIA.

Additionally, I considered to start the construction algorithm of RESPONSE™ CRIA first with a clustering of the customers to get the solution in a promising direction. However, this would be a thesis study in itself and is not done within this thesis research. As can be read above, in RESPONSE™ CRIA I choose to get the solution in a promising direction by running the construction algorithm several times with different search space parameters.

Last but not least, I have considered taking local improvements into account, like the 2-opt or the cross exchange heuristic implemented by Hashimoto *et al.* (2008). As previously mentioned these heuristics need a lot of computation time in comparison to the solution quality. That is, why local improvements are not implemented in RESPONSE™ CRIA.

Notice that in this thesis, the objective of the TDVRP is the same as the objective mentioned in last chapter for solving the VRP with constant speed. That is, I first wish to minimise the number of routes and second to minimise the total travel time over all routes.

4-2 Small Benchmark Problem

This section formulates a small benchmark problem on which the solution method of RESPONSE™ CRIA, defined in the previous section, can be tested. In the literature on the VRP with constant speed, frequently the VRP is tested on the well-know 56 Solomon's benchmark problems formulated by Solomon (1983). These benchmark problems are widely studied and provide an indication of the performance of the algorithm with constant travel speed. Because the TDVRP is an extension of the VRP with constant speed, variations of the Solomon's benchmark problems can be used to test the TDVRP. Figliozzi (2009) has proposed variants of this benchmark that are readily replicable.

The classical set of 56 Solomon's benchmark problems are composed of six different problem types with 100 customers. The six problem classes are named R1, R2, C1, C2, RC1, and RC2. Customers location are randomly generated (problem sets R1 and R2), clustered (problem sets C1 and C2), or mixed with randomly generated and clustered customers (problem sets RC1 and RC2). The customer coordinates are identical for all problems within one type, i.e., R, C and RC. In these problems, travel times equal the corresponding Euclidean distances and travel speeds are 1 kilometre per minute. Problem sets of type 1 have a shorter scheduling horizon, tighter time windows, and a smaller vehicle capacity than problem sets of type 2. Therefore, the solutions of type 2 problems have very few routes and significantly more customers per route.

For testing RESPONSE™ CRIA, I use one of the 56 Solomon's benchmark problems, namely R103. From this benchmark problem, the best known solution to be found in literature, is determined in terms of number of vehicles, travelled distance and the sequence of the orders. Especially the latter is not known for all other benchmark problems. Dataset R103 can be found in Appendix B.

In this chapter, several variants on benchmark problem R103 are taken. The variants are expressed in different scenarios as constructed in Table 4-1. Each scenario differs only in the specific speeds over the network. The scheduling horizon is divided into time-zones of equal durations in the same way as Figliozzi (2009) did. Notice that Scenario 1 takes the original Solomon's problem R103 with constant speed. Scenario 2 and 3 also have constant speed. In Scenario 2 a vehicle moves 20% slower than in Scenario 1 and in Scenario 3 a vehicle moves 20% faster than in Scenario 1. In Scenario 4 and 5, the travel speed of a vehicle depends on the period of the scheduling horizon, these scenarios are in the work of Figliozzi (2009) known as speed distributions TD5 resp. TD2.

Using the proposed benchmark problem within the 5 scenarios, the solution quality of RESPONSE™ CRIA can be tested. By considering only the first 25 customers of the dataset, the structure of the routes can be clarified with the help of pictures. Moreover some conclusions can be made about the advantage of taking into account time-dependent travel times. Next, it is interesting to test RESPONSE™ CRIA on the whole dataset (all 100 customers). In the constant case (that is a simplification of the time-dependent case), the results can now be compared with the best solution ever and in the time-dependent case the results can be compared with the results of the work of Figliozzi (2009). However, as mentioned before, his method does not deal with time-dependent shortest paths calculations (distances between customers are taken as fixed). This is why the computation time of Figliozzi (2009) will be a lot shorter than the computation time of RESPONSE™ CRIA. Anyhow, there is no other literature known with time-dependent benchmarks in which the results of this thesis can be compared. Researchers mentioned in Section 2-2 like Hashimoto *et al.* (2008) and Ichoua *et al.* (2003) uses variations on Solomon's benchmark problems, but none of them uses specific fixed speeds in such a way that their problems can be replicated. In conclusion, to the best of my knowledge, the only work where the results of RESPONSE™ CRIA on the time-dependent scenarios can be compared, is the work of Figliozzi (2009). However, in his paper he presents only the average values over the whole R1 problem class. The values of R103 specifically remain unknown to me.

Table 4-1: Scenarios with travel speed over the time-zones.

Scenario	Number of time-zones	Travel speed (km/min) over the time-zones
1	1	[1.00]
2	1	[0.80]
3	1	[1.20]
4	5	[1.20, 0.80, 1.00, 0.80, 1.20]
5	5	[1.00, 2.00, 1.50, 2.00, 1.00]

4-3 Computational Results

This section shows some computational results on the small dataset and the scenarios mentioned in the previous section. Recall from Section 4-1 that in this theses, the objective of the TDVRP is first to minimise the number of routes and then to minimise the total travel time over all routes. This section first handles the dataset with only the first 25 nodes in Subsection 4-3-1 for constant speed and Section 4-3-2 for time-dependent speed. Next, in Subsection 4-3-3 (that deals with constant speed) and 4-3-4 (that deals with time-dependent speed), the dataset with all 100 customers is considered. This section ends with the computational complexity of RESPONSE™ CRIA in Subsection 4-3-5.

As mentioned in Section 4-1, RESPONSE™ ES stands for the current RESPONSE™ model with the Evolution Strategy. In the computations, there are 200 iterations taken with 100 parents. RESPONSE™ CRIA stands for the Construction and Improvement Algorithm. It is interesting to see the construction of the solution and therefore the construction method from RESPONSE™ CRIA without using the route improvement algorithm is considered. This algorithm will be referred to as RESPONSE™ CA (Construction Algorithm). In the following section on computational results, the method followed by a superscript number that represents the scenario used are given, e.g., RESPONSE™ CRIA^1 means the use of RESPONSE™ CRIA within Scenario 1. Notice that all computation times include the time-dependent shortest path calculations.

4-3-1 Problems with 25 Nodes and Constant Travel Speeds

This subsection presents, the computational results on the problems with 25 nodes and constant speeds (Scenario 1, 2 and 3). The advantage of taking the problems with 25 nodes instead of the problems with 100 nodes are that when the solutions are plotted (or the sequence of orders is given) it is easier to see the results of the several solution methods. Here, the computational results created by RESPONSE^{TM} CRIA will be compared with the results created by RESPONSE^{TM} CA (Construction Algorithm) and the solution created by RESPONSE^{TM} ES. This last method can be applied because the problem in this subsection has only constant travel speeds (there is only 1 time-zone).

The structure of the routes created by RESPONSE^{TM} CRIA will now be clarified using the results of Scenario 1 (the original constant speed). With RESPONSE^{TM} CA a first solution is constructed (Solution 1). The sequence of the orders of Solution 1 is presented in Table 4-2 (this solution is also plotted in Figure C-2 of Appendix C). Here, Solution 2 to 4 describe the solution after each crucial improvement and Solution 5 gives the sequence of the orders of the final best solution calculated by RESPONSE^{TM} CRIA (this solution is also plotted in Figure C-3 of Appendix C). During the creation of Solution 1, the crucial search space parameters are $\{1, 0, 0, 0, 0.3, 0, 0.3\}$ with Order 14 to start the first route. When applying the improvement algorithm, the following critical improvements are done. First to create Solution 2, the smallest route with the two nearest routes are taken (Route 5,4 and 3) and the first route is started with Order 20 and use search space parameters $\{3, 0, 1, 1, 1, 1, 1\}$. Next, to create Solution 3, the second smallest route together with the nearest to this one are taken (Route 4 and 3). Now, the solution can start the first route with Order 7, and taking search space parameter $\{1, 0, 0, 1, 0.3, 1, 0.3\}$ gives the represented solution. Solution 4 is created by taking the smallest route and the second nearest to this smallest route together (Route 3 and 2) and take search space parameters $\{5, 0, 0, 1, 1, 1, 1\}$ and Order 24 to start the first route. The last solution is created by improving the single routes in which by the use of search space parameters $\{5, 0, 0, 1, 1, 1, 1\}$ the sequence of Route 4 is improved as can be seen in Table 4-2.

Table 4-2: Construction of routes using RESPONSE^{TM} CRIA within Scenario 1.

Solution	Route 1	Route 2	Route 3	Route 4	Route 5
1	14 - 15 - 6 - 5 - 16 - 17 - 18	1 - 23 - 22 - 21 - 4 - 25 - 12	2 - 24 - 9 - 20 - 3	7 - 11 - 8 - 19	10 - 13
2	14 - 15 - 6 - 5 - 16 - 17 - 18	1 - 23 - 22 - 21 - 4 - 25 - 12	3 - 24 - 9 - 10 - 19	20 - 11 - 8 - 7 - 13 - 2	
3	14 - 15 - 6 - 5 - 16 - 17 - 18	1 - 23 - 22 - 21 - 4 - 25 - 12	3 - 24 - 9 - 10 - 20	7 - 19 - 11 - 8 - 13 - 2	
4	14 - 15 - 6 - 5 - 16 - 17 - 18	24 - 23 - 22 - 21 - 4 - 25 - 12	1 - 3 - 20 - 9 - 10	7 - 19 - 11 - 8 - 13 - 2	
5	14 - 15 - 6 - 5 - 16 - 17 - 18	24 - 23 - 22 - 21 - 4 - 25 - 12	1 - 3 - 20 - 9 - 10	7 - 19 - 11 - 8 - 2 - 13	

The computational results (in terms of number of routes, total travel time and total distance driven) for Scenario 1, 2 and 3 can be found in Table 4-3. Note that the superscript number i after each solution method indicates the use of the method within Scenario i. As mentioned before, the solution method RESPONSE^{TM} CA^1 gives a solution with five routes and the solution method RESPONSE^{TM} CRIA^1 gives a solution with only four routes. For a first solution, the solution of RESPONSE^{TM} CA^1 is not very bad, but if improvements on this solution are possible, of course this is preferred.

Table 4-3 also shows, that the solution of RESPONSE^{TM} CRIA on Scenario 1 as well as on Scenario 2 and 3, is a little bit worse than the solution of RESPONSE^{TM} ES (same number of routes, but about 1 hour more total travel time). This is quite logical, because RESPONSE^{TM} CRIA is created to handle time-dependent speed problems and can never try as many possible solutions as RESPONSE^{TM} ES can. The solutions of the different methods tested on Scenario 1 can be found in Figure C-1 to C-3 of Appendix C.

Notice that the solution of RESPONSE™ CRIA^2 does not fit within 4 routes any more. Due to the slow speeds, the vehicles need more travel time. Moreover it could happen that the solution will become infeasible, because an order cannot be planned any more. With a speed of 0.80 and the same time windows, Order 36 and Order 65 cannot be planned any more, i.e., when leaving form the depot as early as possible and travelling with a speed of 0.80, customers 65 and 31 can only be served 2 minutes resp. 1 minute after closing time. This is not a problem here, since only the first 25 customers are used.

The solution of RESPONSE™ CRIA^3 is faster in total travel time with the same number of routes. This is not surprising, due to the higher constant speed. Notice that the sequence of the orders of the solution of RESPONSE™ CRIA^3 and RESPONSE™ CRIA^1 is not the same.

All computations for the RESPONSE™ methods in this thesis are done on an Intel Core 2, 2.4 GHz computer. For all RESPONSE™ CRIA computations, the shortest path computation time takes the largest part of the total computation time. For each simulation the total shortest paths computation time is determined and presented in all computational results tables. However, shortest path results from customer to customer at a specific time are stored and not recalculated. The lookup time to see whether shortest path calculation is necessary is not included in the total shortest path computation time. It is clear that shortest path calculations take a significant portion of the total computation time.

Table 4-3: Computational results on R103 with constant speeds and 25 nodes.

Solution method	Number of routes	Total travel time (hours)	Total distance driven (km)	Computation time (h:min:sec)	Of which shortest path (%)
RESPONSE™ ES^1	4	12.48	491	0:1:24	0.06
RESPONSE™ CA^1	5	14.86	596	0:0:04	73.47
RESPONSE™ CRIA^1	4	13.31	513	0:0:08	66.48
RESPONSE™ ES^2	5	14.87	509	0:1:37	0.02
RESPONSE™ CRIA^2	5	16.16	537	0:0:11	53.11
RESPONSE™ ES^3	4	10.98	491	0:1:24	0.06
RESPONSE™ CRIA^3	4	12.09	518	0:0:09	67.28

4-3-2 Problems with 25 Nodes and Time-Dependent Travel Speeds

In this subsection, the computational results on problems with 25 nodes when taking time-dependent travel speeds into account (scenario 4 and 5) are presented. Table 4-4 shows these results.

It is no surprise that the solution from RESPONSE™ CRIA^3 gives a better solution than the one of RESPONSE™ CRIA^4. This is the case, because the lowest speed of Scenario 4 equals the constant speed of Scenario 2.

RESPONSE™ CRIA^5 gives a better solution than RESPONSE™ CRIA^1, which is also obvious, since the lowest speed from Scenario 5 equals the constant speed of Scenario 1.

The whole idea of Scenario 4 is that this time-dependent scenario is more realistic than Scenario 1, that has the same average speed (1 km/min). When a method is used which requires a constant speed, problems may arise in the solution; if we try to insert the solution of RESPONSE™ CRIA^1 into Scenario 4, as can be seen in Table 4-5, the solution does not fit any more, because of the variable travel times. The last order (13) of Route 4 does not fit in the route any more, since the vehicle leaves from Order 2 at 2:43:17, which was 2:37:14 before. Also

the travel times increases from Order 2 to 13 (9.4 minutes to 11.8 minutes) and the closing time of Customer 13 is 2:49:00, hence an additional route is needed.

When a feasible solution is required for a real-life environment and only a time-independent solution method is available, the solution method has to be applied on Scenario 2. Due to the constant lower speed, the solution will always fit into the variable higher speed scenario. However as shown in Table 4-4 this solution is inefficient in comparison with RESPONSE™ CRIA[1]. Notice that the average speed of Scenario 4 equals 1.00 km/min and the average speed of Scenario 2 equals 0.8 km/min. So, when the vehicles travel non-stop during the scheduling horizon, their travel time is 25% longer than in Scenario 4 than in Scenario 2.

Another advantage of having a time dependent solution method is the possibility of finding solutions that are better than the constant speed solution. For instance when the solution of RESPONSE™ CRIA[4] is inserted into Scenario 1, a worse solution is obtained than for RESPONSE™ CRIA[4] itself. Notice that the efficient solution of RESPONSE™ CRIA[4] would never be found by RESPONSE™ CRIA[1]. In some cases the results are even more drastic. When the solution of RESPONSE™ CRIA[4] is inserted into Scenario 2, the solution does not fit any more with the same number of routes (11 routes required instead of 5).

Table 4-4: Computational results on R103 with time-dependent speeds and 25 nodes.

Solution method	Number of routes	Total travel time (hours)	Total distance driven (km)	Computation time (h:min:sec)	Of which shortest path (%)
RESPONSE™ CRIA[1]	4	13.31	513	0:0:08	66.48
RESPONSE™ CRIA[2]	5	16.16	537	0:0:11	53.11
RESPONSE™ CRIA[3]	4	12.09	518	0:0:09	67.28
RESPONSE™ CRIA[4]	4	14.20	545	0:4:06	98.55
RESPONSE™ CRIA[5]	4	11.17	565	0:3:48	98.80

Table 4-5: Computational results on dataset R103 with 25 nodes, the sequence of the orders of the calculated solutions using the speeds of different scenarios.

Solution calculated by	Inserted in scenario	Number of routes	Total travel time (hours)	Total distance driven (km)
RESPONSE™ CRIA[1]	4	5	16.37	532
RESPONSE™ CRIA[2]	4	5	15.64	537
RESPONSE™ CRIA[4]	1	4	14.23	545
RESPONSE™ CRIA[4]	2	11	27.77	777

4-3-3 Problems with 100 Nodes and Constant Travel Speeds

This subsection presents the computational results on the problem with 100 nodes and constant travel speeds, i.e., the original Solomon's benchmark problem. The advantage of testing the method on these 100 nodes problems instead of the 25 nodes problems is that solutions for these Solomon's benchmark problem are known for several solution methods presented in the literature. For dataset R103, the best known solution in terms of number of vehicles and travelled distances is presented order to order by Gambardella (1999). From this information, the best known solution of R103 is plotted in Figure C-4 of Appendix C.

The computational results on the problems with 100 nodes and constant travel speeds are presented in Table 4-6. Notice that for the best ever solution the primary objective is to minimise the number of vehicles and the secondary objective is to minimise the shortest distance. This also is the objective that Figliozzi (2009) uses for the constant speed problems. However for RESPONSE™ CRIA the total travel time is used as secondary objective, since this is part of the

goal of this thesis. This leads to the solutions formulated in Table 4-6. For more information on the best ever solution the reader is referred to Hasle (2009). In the table the results of Figliozzi (2009) are also presented. It must be noted that these are given in an average value over the whole R1 problem class (12 scenarios). Unfortunately the specific results of R103 are unknown to me.

The solution of RESPONSE™ CRIA comes close to the best ever solution (14 routes vs 13), however there is a large computational cost. Every route improvement takes 8 minutes calculation time and on average 8 route improvements are needed in 4 loops (this loop is presented in Appendix A-6).

For illustrative purposes the solutions of RESPONSE™ CRIA and Best Ever on dataset R103 for Scenario 1 are given in Figure C-5 to C-4 of Appendix C.

Table 4-6: Computational results on R103 with constant speeds and 100 nodes.

Solution method	Number of routes	Total travel time (hours)	Total distance driven (km)	Computation time (h:min:sec)	Of which shortest path (%)
Best Ever^{1,average}	11.92	-	1,210	-	-
Figliozzi (2009)^{1,average}	12.58	-	1,248	0:0:20	0.00
Best Ever^{1}	13	45.49	1,293	-	-
RESPONSE™ ES^{1}	14	38.96	1,331	0:10:14	0.03
RESPONSE™ CRIA^{1}	14	44.64	1,588	0:51:59	28.01

4-3-4 Problems with 100 Nodes and Time-Dependent Travel Speeds

This subsection discusses the computational results on problems with 100 nodes and time-dependent travel speeds. Table 4-7 shows these results. The superscript *4 soft* denotes Scenario 4 using soft time-windows. Where soft means the soft time-window violation principle proposed by Figliozzi (2009): $P_{max} = 0.1(l_0 - e_0) = e_i^{\#} - e_i = l_i - l_i^{\#}$, with $[e_i, l_i]$ the hard time window interval and $[e_i^{\#}, l_i^{\#}]$ the soft time window interval and where $i = 0$ equals the depot. Notice that the depot working time $[e_0, l_0]$ is not relaxed. Scenario 5 does not require soft time windows, since a feasible solution can always be found, due to the low speed.

Notice that the computation time of Figliozzi (2009) is calculated without the shortest path algorithm and averaged over 65 problems using a Dell Latitude 430, 1.2 GHz computer. The solutions of RESPONSE™ are calculated on an Intel Core 2, 2.4 GHz computer and include the time-dependent shortest path calculations.

Comparing the results of Figliozzi (2009) with the results of RESPONSE™ CRIA leads to the believe that Figliozzi (2009) presents his travel time without waiting time. This is denoted in Table 4-7 as the effective travel time. Notice that the secondary objective of Figliozzi (2009) for problems with time-dependent speeds, is this effective travel time. It seems the results of RESPONSE™ CRIA are of reasonable quality, however the computational cost is a problem. The largest part of this problem stems from the use of the shortest path algorithm in RESPONSE™ CRIA (about 97% of the total computation time).

Table 4-7: Computational results on R103 with time-dependent speeds and 100 nodes.

Solution method	Number of routes	Total travel time (hours)	Effective travel time (hours)	Total distance driven (km)	Computation time (h:min:sec)	Of which shortest path (%)
Figliozzi (2009)$^{4\ soft,average}$	10.42	-	18.90	1,131	0:0:21	0.00
RESPONSE™ CRIA$^{4\ soft}$	11	39.26	22.30	1,349	49:20:34	98.08
Figliozzi (2009)^{5,average}	10.75	-	14.95	1,258	0:0:19	0.00
RESPONSE™ CRIA^{5}	12	37.82	19.31	1,679	22:27:17	96.51

4-3-5 Computational Complexity

This subsection describes the runtime complexity of RESPONSE™ CRIA. As mentioned in Subsection 4-1-3, RESPONSE™ CRIA uses the construction algorithm multiple times. This construction algorithm is called during the TDVRP process no more than $n_s \cdot \delta \cdot S$ times. Where S is the total number of order subsets, n_s is the number of customers tried to start the route, and δ is the number of search space parameters. Notice that n_s, δ and S are finite numbers and S depends on when the proposed algorithms have converged. From this I can conclude that the computational complexity of the construction algorithm has a substantial impact on the running time of the TDVRP process.

The key parameters in terms of computation time are the number of customers and the number of time zones. First the impact of the number of customers will be investigated. For this, instances with different number of customers have to be run. When running only the start solution (using RESPONSE™ CA), n_s, δ and S have the same constant values when varying the number of customers, which forms a good basis of comparison. Note that, S equals one, the complete order list.

Table 4-8 shows the results of RESPONSE™ CA on the Solomons problem R103 with constant speed. Solomon's problem is defined with $n = 100$, $n = 25$ and $n = 50$ are defined using a selection of R103 and $n = 200$ and $n = 250$ are defined using copies of the customers with new coordinates. Notice that the results are expressed as the ratio between each computation time and the computation time for $n = 25$. The last column of the table shows that as the number of customers n increases, the ratio as a percentage of the n^3 growth factor is decreasing. From this, I conclude that the running time is increasing by a factor of $O(n^2)$. This is expected, because Figliozzi (2009) made the same conclusion for a similar construction algorithm.

Now that the complexity of RESPONSE™ CA is determined to be of order $O(n_s \cdot \delta \cdot S \cdot n^2)$. Then, from the structure of RESPONSE™ CRIA I observe that the impact on this solution method is similar. The only difference lies in S, which is 1 for RESPONSE™ CA and dependent on the specific problem for RESPONSE™ CRIA.

Secondly, the increase in computation time dependent on the number of time-zones is checked. The results are expressed in Table 4-9, in which p indicates the number of time-zones. From this it can be obtained that the computation time increases by a factor of $O(p)$.

The major contribution of computation time due to increase in time-zones is caused in the shortest path algorithm. When calculating the shortest path using the algorithms presented in Appendix A-1 and A-2, an increase in time-windows will trigger the function CrossTimeZone-High more often.

As mentioned in Section 2-2, researchers sometimes run into memory problems of their computers. Their problems occur during pre-calculation of the shortest path matrix. Since RESPONSE™ CRIA does not use such a matrix, this problem is not experienced. However similar problems may occur during the TDVRP process. The reason for this is that the shortest path results from customer to customer at a specific time are stored during the TDVRP process, in order to prevent recalculation when the same shortest path is needed again. This storage space also becomes a critical factor for a large number of customers. For problems like the one presented in Subsection 4-3-4 with 100 customers and 5 time-zones, memory limitations are reached after many iterations. Clearing the memory of the shortest path results is needed, which of course causes a setback in computation time. An advantage is that clearing the memory of the shortest path results is possible, something which would not be the case during a pre-calculation phase.

In conclusion experimental results indicate that the computation time increases proportionally to the number of time-zones, as well as to the square of the number of customers.

Table 4-8: Computation time ratio depending of the number of customers

Number of customers (n)	$O(n^2)$	$O(n^3)$	Ratio	% $O(n^3)$
25	1	1	1.0	100
50	4	8	3.9	49
100	16	64	17.5	27
200	64	512	100.5	20
250	100	1,000	174.0	17

Table 4-9: Computation time ratio depending of the number of time-zones

Number of time-zones (p)	$O(p)$	$O(p^2)$	Ratio	% $O(p^2)$
2	1	1	1.0	100
4	2	4	3.1	77
8	4	16	7.4	46
16	8	64	18.0	28

Chapter 5

Time-Dependency related to a Real-Life Problem

This chapter describes a real-life problem. First the real-life dataset is defined on which the solution method of RESPONSE^{TM} CRIA, presented in Section 4-1, can be tested. The chapter ends with the computational results.

In the previous chapter, the travel speed depended only on the time-zone. In other words the travel speed at a given time was constant over the entire network. In real life the travel speed is not only time-dependent, but also position dependent. This is because, there are several types of roads in the network (highway, trough way, main road, local road), each of which has its own travel speed, and where congestion of the network is local (not every highway is congested at once).

The real-life problem contains a part of The Netherlands, i.e., the urban agglomeration that stretches from Rotterdam to Amsterdam with a surface of about 60 km^2. The network, which contains information on road type, free-flow speed and off-peak hours speed of a truck for every link, is derived from a dataset that TNO BU M&L owns. Here the nodes are given in RD coordinates (Rijks Driehoek coordinates). The network of the real-life problem used contains 962 links, which is small compared to the network that RESPONSE usually uses of West Europe, 10,000 links. In these networks however, only the main roads are modelled, and many more links are needed if also the small roads would be added to the model. Note that the Solomon's problem of the previous chapter uses a network of 10,100 links, since every customer can be reached on one link from every customer. I choose the number of customers of the real-life problem used to be 35, for visualisation purposes. This is nothing compared to a large transportation company in the Netherlands, that delivers 1,000 orders per day.

Collecting the required time-dependent travel speed information is a very time consuming. Even if the global properties of the travel time variation are known, for example, rush hour periods in the early morning and in the late afternoon, the single links in the road network may be affected quite differently.

In a recent development at TNO, time dependent road maps made by the American company INRIX have become available. In these road maps the travel speeds are recorded for many links in the network at different times during the day. INRIX and TNO have a combined effort in producing models that can predict traffic behaviour in Europe. Also the ANWB travel time

predictor uses the models made by INRIX, in which actual traffic jam information is linked to TMC coordinates (Traffic Message Channel coordinates). If the TMC coordinates could be converted to RD coordinates (or any other geographical system), then it would be possible to use this network in RESPONSE^{TM} CRIA. This would have a very large potential benefit for TNO. However at this moment there is no working conversion algorithm and it would be a thesis in itself to create one.

Since a conversion algorithm does not (yet) exist, the choice has been made to simulate the travel speed variations. Of course this should be done in the most realistic way. Therefore the congestion of a Thursday (a normal work day) is simulated with the time-zones as presented in Table 5-1. Here, the speeds over the network during the morning rush hours and evening rush hours are modelled with the assumption that during the evening rush hours on highways and through ways, there is a little bit more congestion than during the morning rush hours. On the other hand, on roads where a lower maximum speed is allowed, the morning rush hours are a little bit worse than the evening rush hours. To simulate position dependent congestion, a random factor is added. Notice that all speeds are simulated for a truck that delivers the orders with a maximum speed of 80 km/hour as defined by national law.

In mathematical formulation, take:

V	the simulated speed on a link in this time-zone (km/hour);
V_{max}	the maximum speed on a link (km/hour);
$V_{\text{off-peak}}$	the off-peak hours speed on a link (km/hour);
γ_{link}	a random number between 0 And 1 for each link.

Then, the speeds over the network for Time-zone 2 are simulated as:

If $V_{\text{max}} > 90$ then
$V = 0.2 \cdot V_{\text{off-peak}} + \gamma_{\text{link}} \cdot 0.8 \cdot V_{\text{off-peak}}$
Else
$V = 0.85 \cdot V_{\text{off-peak}} + \gamma_{\text{link}} \cdot 0.15 \cdot V_{\text{off-peak}}$.

On a same way the speeds over the network for Time-zone 4 are simulated as:

If $V_{\text{max}} > 90$ then
$V = 0.15 \cdot V_{\text{off-peak}} + \gamma_{\text{link}} \cdot 0.85 \cdot V_{\text{off-peak}}$
Else
$V = 0.89 \cdot V_{\text{off-peak}} + \gamma_{\text{link}} \cdot 0.11 \cdot V_{\text{off-peak}}$.

The real-life dataset can be found in Appendix B-1. Notice that the depot is open the whole day (24 hours), but the customers need to be delivered within the time-windows.

Table 5-1: Time-zones for the real-life problem

Time-zone nr.	Traffic situation	Start time	End time
1	Off-peak hours	00:00:00	07:30:00
2	Morning rush hours	07:30:00	09:15:00
3	Off-peak hours	09:15:00	16:45:00
4	Evening rush hours	16:45:00	18:45:00
5	Off-peak hours	18:45:00	24.00:00

To present the difference between RESPONSE™ CRIA and a non time-dependent method, a solution of the real-life problem using 5 time-zones is compare to a solution using 1 time-zone with the average speed per link. Figure C-6 of Appendix C-3 shows the solution with one time-zone and C-7 shows the solution with 5 time-zones. Table 5-2 shows the computational results of both, with only five trucks all 35 customers can be delivered on time. It must be noted that the time-dependent method clearly is more efficient. When the single time-zone solution is inserted into the 5 time-zone environment, a solution of 4 routes with 33 hours travel time is found. However when the 5 time-zone solution is inserted into the single time-zone environment, the solution becomes infeasible and more routes are needed. This means that the 5 time-zone solution, of 3 routes with 31 hours travel time, would never be found by the non time-dependent method.

Table 5-2: Computational results on real-life problem.

Solution method	Nr of routes	Total travel time (hours)	Total distance driven (km)	Computation time (h:min:sec)	Of which shortest path (%)
RESPONSE™ CRIA^{1TZ}	4	33.17	995	0:0:05	77.19
RESPONSE™ CRIA^{5TZ}	3	31.99	957	0:0:13	95.15

Table 5-3: Computational results on real-life problem, the sequence of the orders of the calculated solutions using the speeds of different scenarios.

Solution calculated by	Inserted in scenario	Number of routes	Total travel time (hours)	Total distance driven (km)
RESPONSE™ CRIA^{1TZ}	5 Time-Zones	4	33.61	995
RESPONSE™ CRIA^{5TZ}	1 Time-Zone	5	37.40	1,068

Figure C-8 of Appendix C-3 shows the travelled links for the single time-zone solution and Figure C-9 shows travelled links for the solution with 5 time-zones. The preference for highways is clear for both, but for the 5 time-zones solution also the avoidance of congested areas can be seen. A part of the Dutch highways A1 and A4 are avoided and secondary roads, which are less congested are used.

Chapter 6

Conclusions and Recommendations

This chapter describes the conclusions and recommendations of this thesis. Section 6-1 formulates conclusions based on the research project and Section 6-2 presents recommendations for future work.

6-1 Conclusions

The objective of this thesis was defined in Section 1-2 as developing an efficient method that produces good quality solutions to real-life Vehicle Routing Problems with time-dependent travel times. In this thesis a literature study on the Vehicle Routing Problem (VRP) has been done in which VRP variants and solution methods are formulated that structured the field of the VRP. And also a literature review on the Time-Dependent Vehicle Routing Problem (TDVRP) is made. From this review I was inspired to design and implement my Construction and Route Improvement Algorithm (CRIA), which can handle time-dependent travel times. The algorithm will be discussed in light of realism, efficiency, followed by a general conclusion.

Realism The developed CRIA is integrated with a time-dependent shortest path algorithm that will be executed during the TDVRP process. No other published work using this methodology is found. In all research on the TDVRP, one or a set of paths are stored before the start of the TDVRP calculation, and during the TDVRP process a pre-calculated path will be selected and the time-dependent travel time corresponding to the path will be calculated. A drawback of this methodology with varying speeds, but constant distances, is that it does not consider (enough) potential changes of the shortest (i.e., fastest) paths themselves. Due to varying travel times, alternative paths (with different distances) may result in a shorter path between customers. The methodology used in this thesis therefore gives a more realistic solution, that is in line with the objective of this thesis. Routes will be created that avoid locations with congestion during rush hours.

In any way, a solution created with a time-dependent method is more realistic than one created with a method that can only handle constant travel times. A solution obtained with constant speeds may well be infeasible in the real-life time-dependent context, i.e., the closing time at a customer or at the depot may be exceeded. Moreover there is a possibility to find solutions that are better than the constant speed solutions in the time-dependent context, that otherwise would never have been found. Notice that in this thesis a visualisation is made for the performance of a real-life problem. This visualisation shows that congested areas are indeed avoided.

Efficiency Because local changes of a route do not only influence the travel times around a few exchanged customers, but may have impact on all other customers on the route, local improvement heuristic have to be avoided. Therefore and due to the resulting big influence on computation time, the developed CRIA takes a route improvement approach instead of the local improvement approach that most researchers take. The developed CRIA is tested on one of the Solomon's benchmark problems and works quite good. For constant scenarios, the methods created for dealing with constant scenarios perform better though. However, taking into account that the developed CRIA is created for time-dependent purposes, the results can be said to have reasonable quality. Because there are only a few people that did research on the TDVRP and their experimental results cannot easily be compared, it is not possible to compare the quality of the time-dependent cases. The computational cost becomes problematic for large instances (more than 100 customers and 5 time-zones). The largest part of this problem stems from the use of the time-dependent shortest path algorithm. Because the solver typically can be started some time before the start of the planning horizon, the computation time is not a critical factor and this drawback is of less importance for problems up to 100 customers.

Notice that experimental results from this thesis indicate that the computation time of the time-dependent solution method increases proportionally to the number of time-zones, as well as to the square of the number of customers. Thus for the TDVRP, the developed CRIA can be referred to as an relatively efficient solver.

In conclusion In this thesis, complex optimisation algorithms are explored, from which an efficient method is created that handles time-dependent travel times and produces good quality solutions for real-life problems. Comparison between this new method and existing methods is difficult, but the solutions of the method seem to have good quality. The advantages are clear: the possibility to generate good quality solutions of the TDVRP that better reflect reality.

6-2 Recommendations

This section presents some recommendations for future work. The efficient solution method, presented in this thesis gives realistic solutions when dealing with time-dependent travel times. Similar to the solution method for the classic VRP, it deals with several boundary conditions. Therefore, further research can be done by handling additional assumptions, like multiple time windows per customer/depot, driving hours regulations for truck drivers or a heterogeneous fleet of vehicles. For all these additional assumptions a trade off should be made between increase in realism and additional computation time.

Due to the time-dependency, the created routes cannot always be shifted over time. Where in the constant scenarios the created routes do not always have a fixed begin or end time, in the time-dependent scenarios they have. That is why in the time-dependent context a route to vehicle allocation problem is not relevant. Due to this complexity, the developed CRIA is not yet able to solve a TDVRP with multiple routes per vehicle. Therefore, the vehicle routes that are full because of capacity constraints may become suboptimal. Further research can be done on the TDVRP with multiple routes per vehicle. Notice that the cost of an extra truck becomes important in this case. The consideration has to be made, when it is profitable to let a truck do a next route from the depot with the possibility that less orders can be serviced, due to closing time-windows, or when an extra truck should be used, which can start from the depot much earlier.

As can be read in this thesis, the insertion strategy of the developed CRIA is dependent on several search space parameters. Here, the most promising parameters are taken, but obviously an extended research can be done to analyse the efficiency of these parameters. Moreover more research can be done on the efficiency of the route-improvements, i.e., make other combinations of the subsets of routes.

Further research can be to add multiple time-zones to the Facility Location Model that handles strategic problems. It is a challenge to model the congestion also in this tool, such that an strategic optimisation can be done by taking into account the time-dependent optimal routes calculated by the VRP.

There will be random variations in travel times, mainly due to accidents or weather conditions, that are unpredictable. These variations can and should be considered in dynamic vehicle routing, where orders arrive during the planning horizon and decisions are taken in real-time based on the current state of information. Therefore, it is recommended to extend the developed CRIA with the ability to handle the Dynamic Vehicle Routing Problem (DVRP). Note that due the long computation time it can be advised that the number of shortest path calculations is reduced, since this is a critical factor in the computation time. The rationale behind this optimisation is that in a dynamic reality, a truck can decide to leave earlier or later to avoid periods of high congestion.

There is an opportunity for TNO, to bring out a broader scale of advice by taking the influence of road-pricing into account. In a recently proposed law by minister Eurlings from Verkeer en Waterstaat (2009) kilometre-pricing for freight transportation will be introduced gradually from 2012 until 2018. During this transition phase the kilometre-price will increase several times. This kilometre-pricing is expected to become location and time dependent. Which roads will be expensive at what time is currently unknown. With the developed CRIA, time-dependent calculations can be done and with some adjustments it should also be able to deal with location dependency. Then, the objective can be changed from optimising on shortest travel time to optimising on minimal kilometre cost. And also, the shortest path algorithm can be adapted to a cheapest path algorithm, where the input becomes the kilometre-pricing per link.

As a last remark, when time-dependent measurement data connected to a geographic network would become available, solutions would be even more realistic. By creating a link between the measurement data and the developed Construction and Route Improvement Algorithm, the realism and usefulness of the solutions would increase substantially.

Hopefully, the results presented in this thesis will contribute to the effectiveness and the efficiency of future projects in dynamic planning in logistics.

Appendix A

Implementation Codes

- Confidential -

Appendix B

Datasets

B-1 Solomon's Dataset R103

Table B-1: Solomon's dataset R103 with vehicle capacity 200 and cust no. 0 equal to the depot.

cust no.	xcoord.	ycoord.	demand	ready time (min)	due date (min)	service time (min)
0	35	35	0	0	230	0
1	41	49	10	0	204	10
2	35	17	7	0	202	10
3	55	45	13	0	197	10
4	55	20	19	149	159	10
5	15	30	26	0	199	10
6	25	30	3	99	109	10
7	20	50	5	0	198	10
8	10	43	9	95	105	10
9	55	60	16	97	107	10
10	30	60	16	124	134	10
11	20	65	12	67	77	10
12	50	35	19	0	205	10
13	30	25	23	159	169	10
14	15	10	20	0	187	10
15	30	5	8	61	71	10
16	10	20	19	0	190	10
17	5	30	2	157	167	10
18	20	40	12	0	204	10
19	15	60	17	0	187	10
20	45	65	9	0	188	10
21	45	20	11	0	201	10
22	45	10	18	97	107	10
23	55	5	29	68	78	10
24	65	35	3	0	190	10
25	65	20	6	172	182	10
26	45	30	17	0	208	10
27	35	40	16	37	47	10
28	41	37	16	0	213	10
29	64	42	9	0	190	10
30	40	60	21	71	81	10
31	31	52	27	0	202	10
32	35	69	23	0	186	10
33	53	52	11	37	47	10
34	65	55	14	0	183	10
35	63	65	8	143	153	10
36	2	60	5	41	51	10
37	20	20	8	0	198	10
38	5	5	16	83	93	10
39	60	12	31	44	54	10

cust no.	xcoord.	ycoord.	demand	ready time (min)	due date (min)	service time (min)
40	40	25	9	85	95	10
41	42	7	5	97	107	10
42	24	12	5	31	41	10
43	23	3	7	0	185	10
44	11	14	18	69	79	10
45	6	38	16	32	42	10
46	2	48	1	0	184	10
47	8	56	27	0	185	10
48	13	52	36	0	192	10
49	6	68	30	108	118	10
50	47	47	13	0	203	10
51	49	58	10	0	193	10
52	27	43	9	0	208	10
53	37	31	14	95	105	10
54	57	29	18	0	197	10
55	63	23	2	136	146	10
56	53	12	6	130	140	10
57	32	12	7	0	196	10
58	36	26	18	200	210	10
59	21	24	28	0	202	10
60	17	34	3	0	201	10
61	12	24	13	0	194	10
62	24	58	19	58	68	10
63	27	69	10	0	185	10
64	15	77	9	73	83	10
65	62	77	20	51	61	10
66	49	73	25	127	137	10
67	67	5	25	83	93	10
68	56	39	36	142	152	10
69	37	47	6	50	60	10
70	37	56	5	182	192	10
71	57	68	15	0	180	10
72	47	16	25	0	197	10
73	44	17	9	0	199	10
74	46	13	8	149	159	10
75	49	11	18	0	192	10
76	49	42	13	73	83	10
77	53	43	14	179	189	10
78	61	52	3	96	106	10
79	57	48	23	92	102	10
80	56	37	6	182	192	10
81	55	54	26	0	192	10
82	15	47	16	0	196	10
83	14	37	11	0	198	10
84	11	31	7	101	111	10
85	16	22	41	0	196	10
86	4	18	35	0	184	10
87	28	18	26	93	103	10
88	26	52	9	74	84	10
89	26	35	15	0	211	10
90	31	67	3	0	187	10
91	15	19	1	0	194	10
92	22	22	2	18	28	10
93	18	24	22	188	198	10
94	26	27	27	0	207	10
95	25	24	20	0	205	10
96	22	27	11	0	204	10
97	25	21	12	0	202	10
98	19	21	10	0	198	10
99	20	26	9	83	93	10
100	18	18	17	185	195	10

B-2 Real-Life Dataset

Table B-2: Real-life dataset with vehicle capacity 1,000 and cust no. 0 equal to the depot.

cust no.	settlement	xcoord.	ycoord.	demand	ready time (h:min)	due date (h:min)	service time (min)
0	Rotterdam-Haven	92260	433887	0	00:00	24:00	60
1	Papendrecht	106952	428157	90	08:00	12:00	24
2	Leerdam	134426	434622	60	08:00	12:00	21
3	Rotterdam	94051	438588	75	08:00	12:00	23
4	Schoonhoven	117942	440205	50	08:00	12:00	20
5	Nieuwerkerk aan den IJssel	102005	442832	50	08:00	12:00	20
6	Vianen	135160	444906	60	08:00	12:00	21
7	Delft-Zuid	83885	445788	75	08:00	12:00	23
8	Gouda	108274	447257	90	08:00	12:00	24
9	Pijnacker	89468	447697	50	08:00	12:00	20
10	IJsselstein	131655	448522	50	08:00	12:00	20
11	Delft-Noord	84326	449167	90	08:00	12:00	24
12	Montfoort	125317	449167	50	08:00	12:00	20
13	Waddingsveen	104631	451913	50	08:00	12:00	20
14	Zoetermeer	94317	452840	110	08:00	12:00	26
15	Bodegraven	112947	453868	50	08:00	12:00	20
16	Woerden	119881	454456	50	08:00	12:00	20
17	Den Haag	80359	454750	110	08:00	12:00	26
18	Utrecht	137511	456366	125	08:00	12:00	28
19	Wassenaar	84914	458863	60	08:00	12:00	21
20	Maarssen	130233	460994	85	13:00	17:00	24
21	Nieuwkoop	113122	461067	50	13:00	17:00	20
22	Leiden	92133	462329	85	13:00	17:00	24
23	Alpen aan den Rijn	106511	462536	75	13:00	17:00	23
24	Hilversum	141478	468707	75	13:00	17:00	23
25	Mijdrecht	117677	468854	50	13:00	17:00	20
26	Uithoorn	117236	472674	60	13:00	17:00	21
27	Lisse	97255	474731	60	13:00	17:00	21
28	Aalsmeer	111506	475025	60	13:00	17:00	21
29	Bussum	138686	475906	60	13:00	17:00	21
30	Amstelveen	119293	478991	90	13:00	17:00	24
31	Hoofddorp	105189	480167	75	13:00	17:00	23
32	Amsterdam-Oost	126198	482664	110	13:00	17:00	26
33	Heemstede	105483	486044	90	13:00	17:00	24
34	Haarlem	100346	486718	100	13:00	17:00	25
35	Amsterdam-West	116795	486778	90	13:00	17:00	24

Appendix C

Plots of Solutions

C-1 Solutions of R103 with 25 Nodes

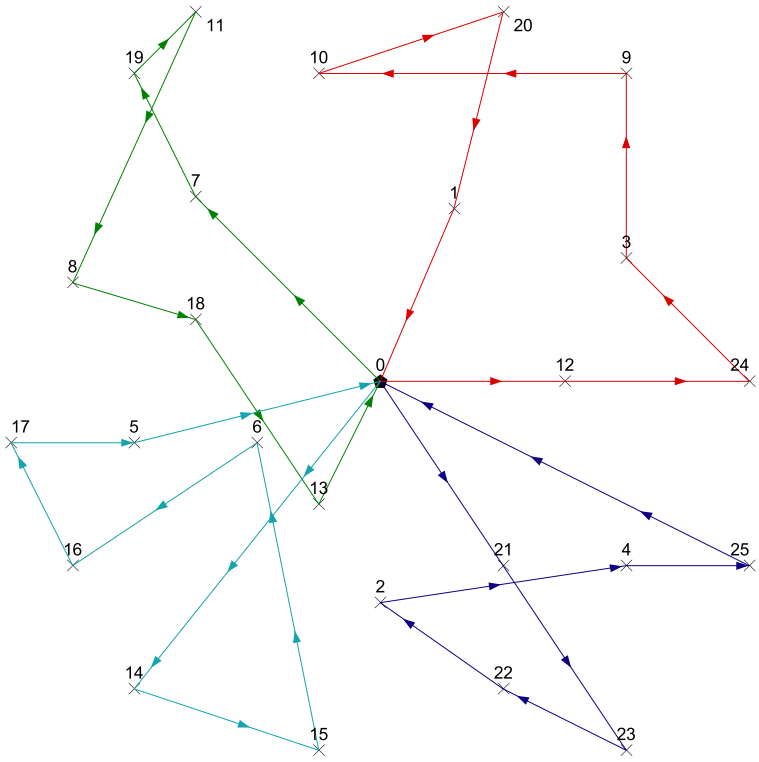

Figure C-1: The solution of R103 with 25 nodes created by RESPONSE™ ES[1].

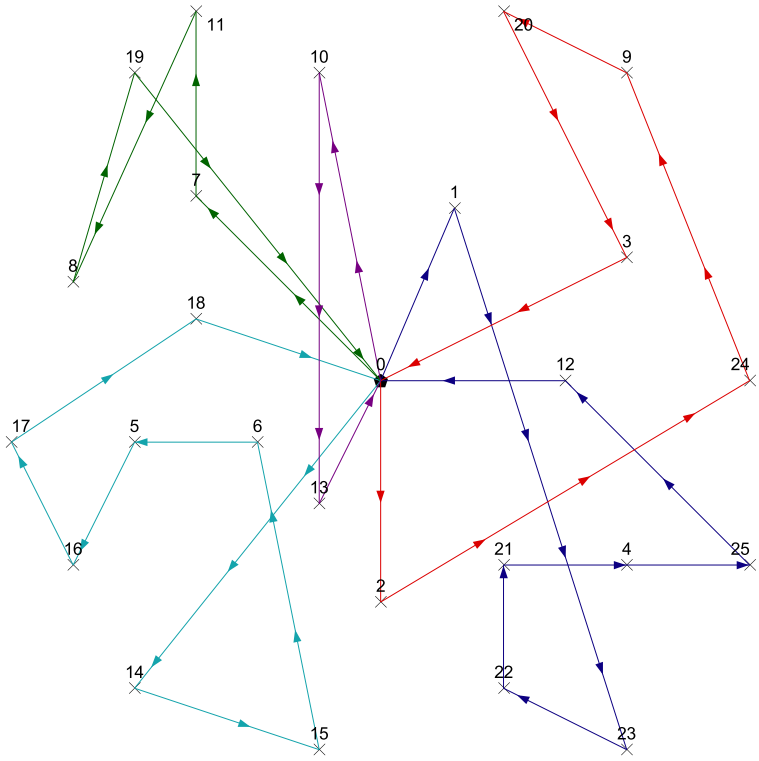

Figure C-2: The solution of R103 with 25 nodes created by RESPONSE™ CA[1].

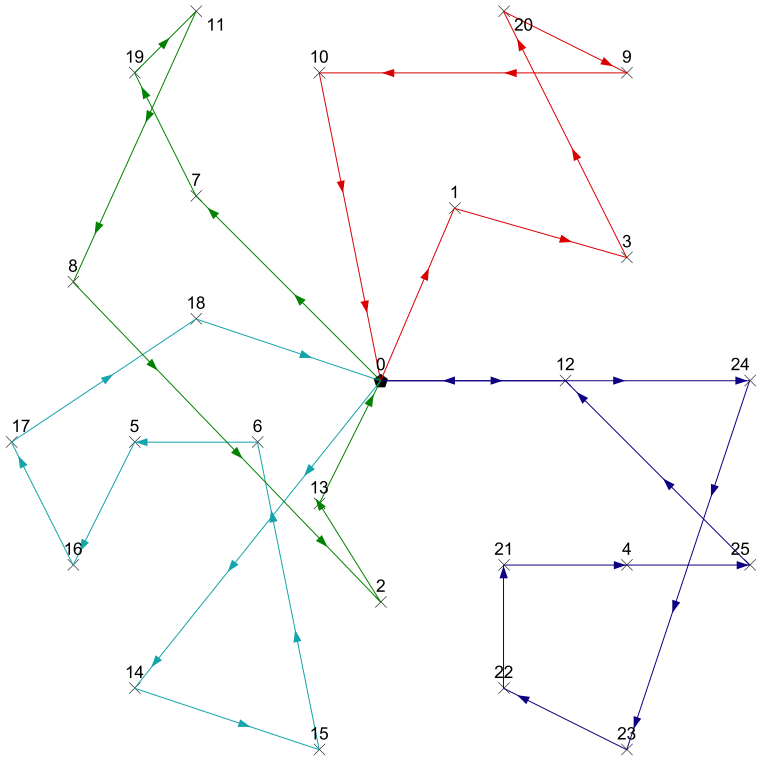

Figure C-3: The solution of R103 with 25 nodes created by RESPONSE™ CRIA[1].

C-2 Solutions of R103 with 100 Nodes

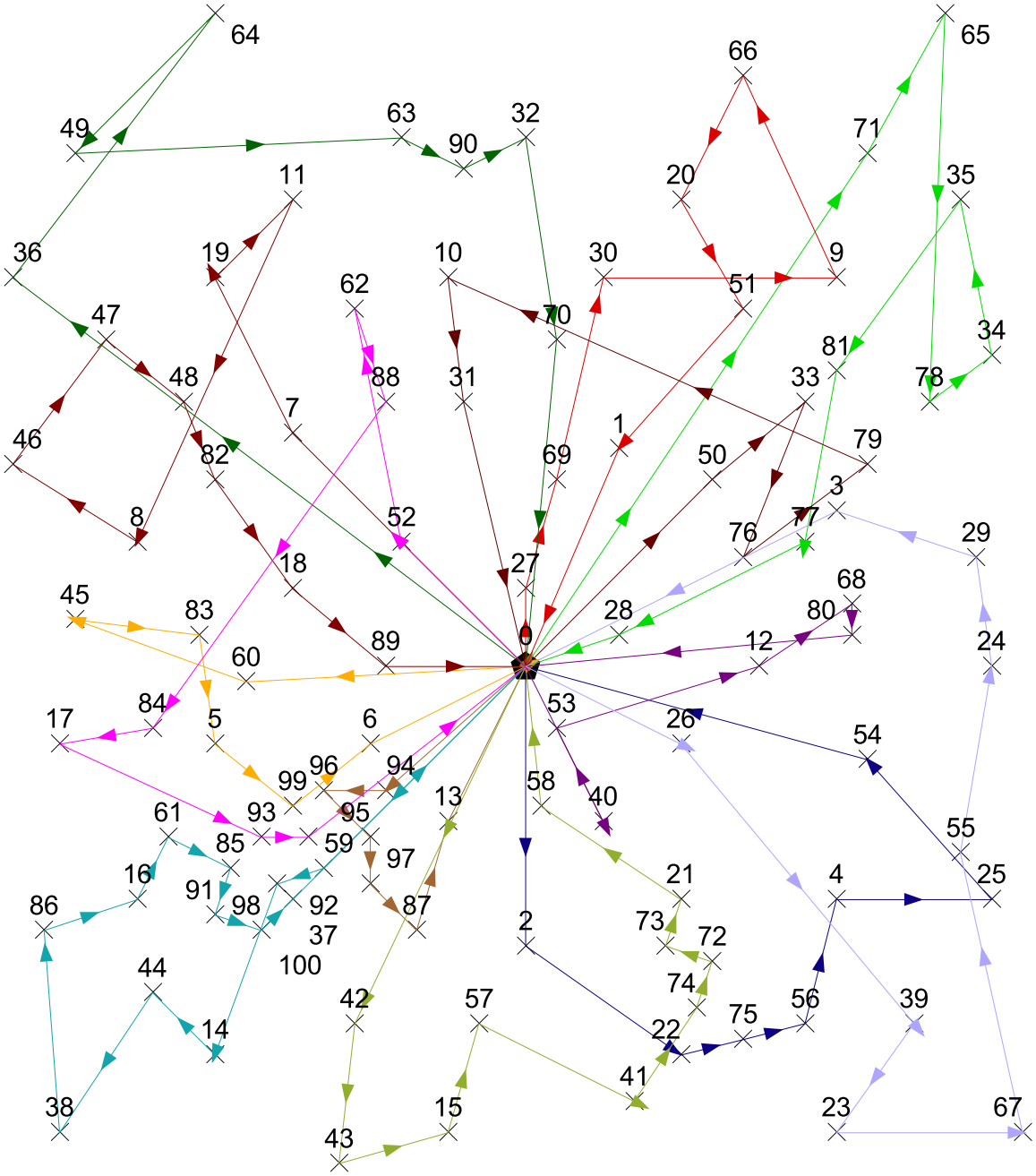

Figure C-4: The best known solution of R103 with 100 nodes.

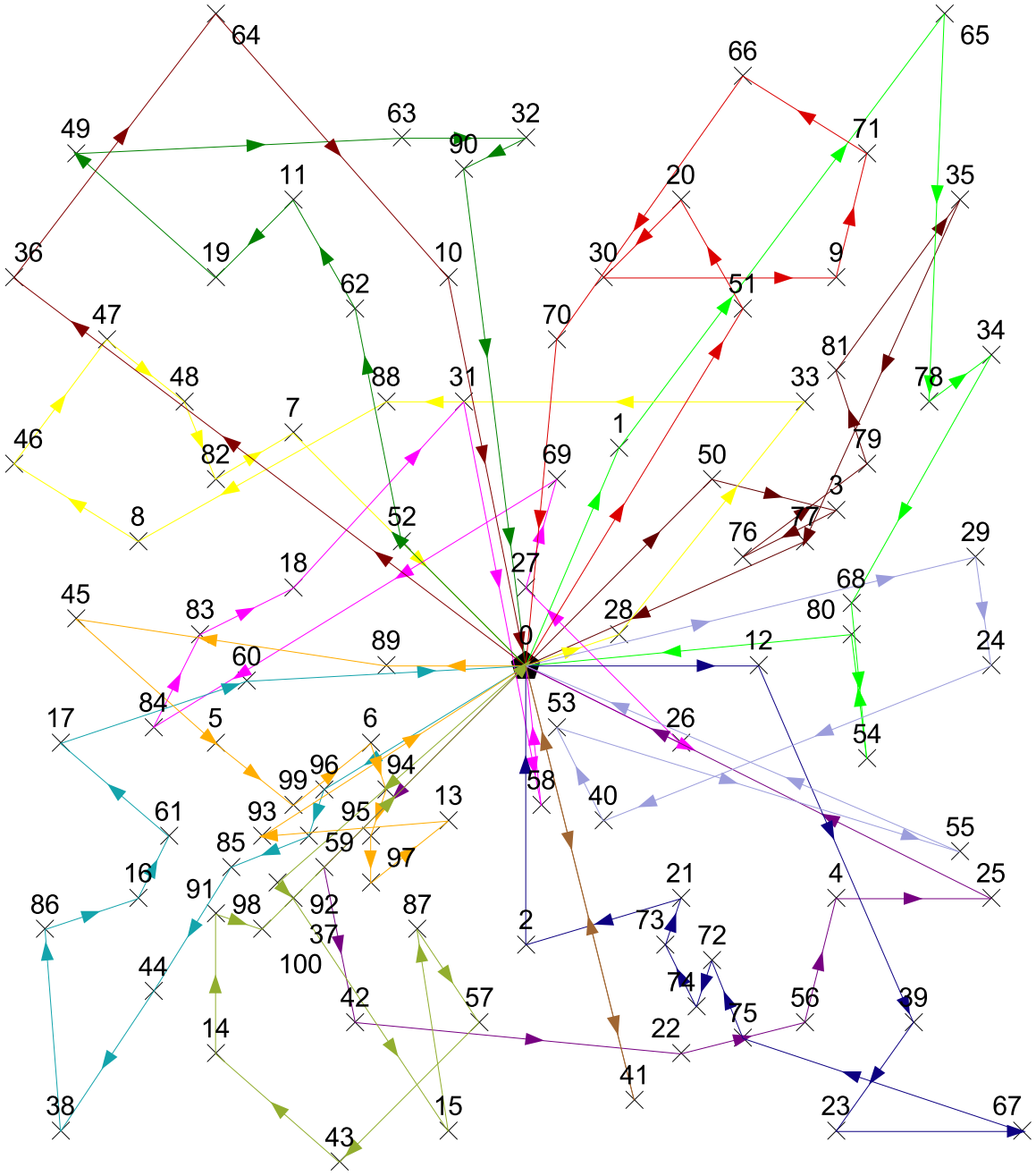

Figure C-5: The solution of R103 with 100 nodes created by RESPONSE^{TM} CRIA.

C-3 The Solution of the Real-Life Problem

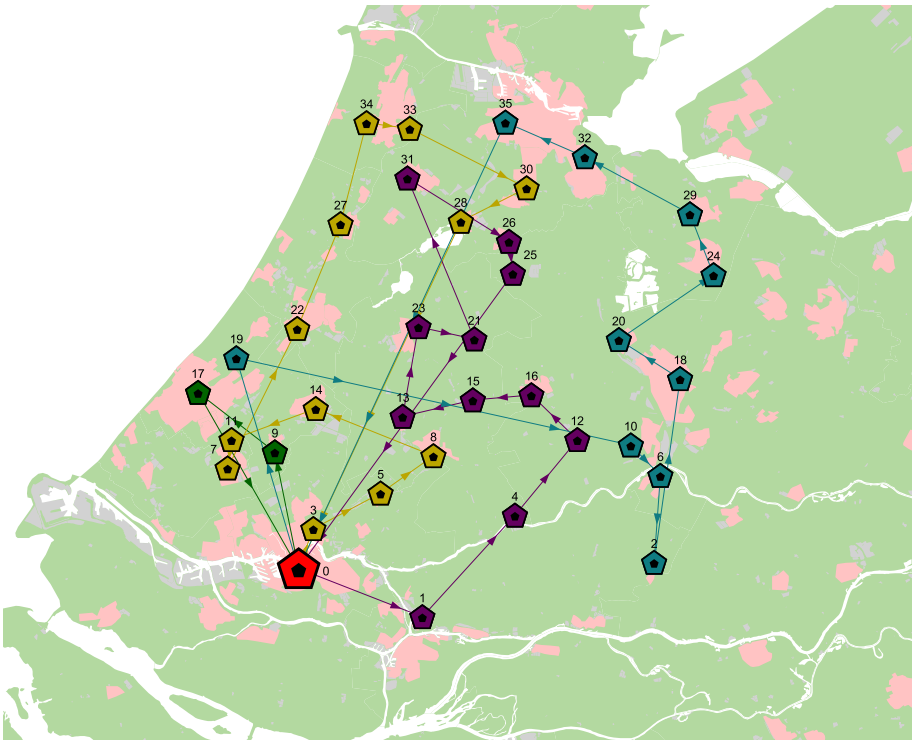

Figure C-6: The solution of the real-life problem taking one time-zone

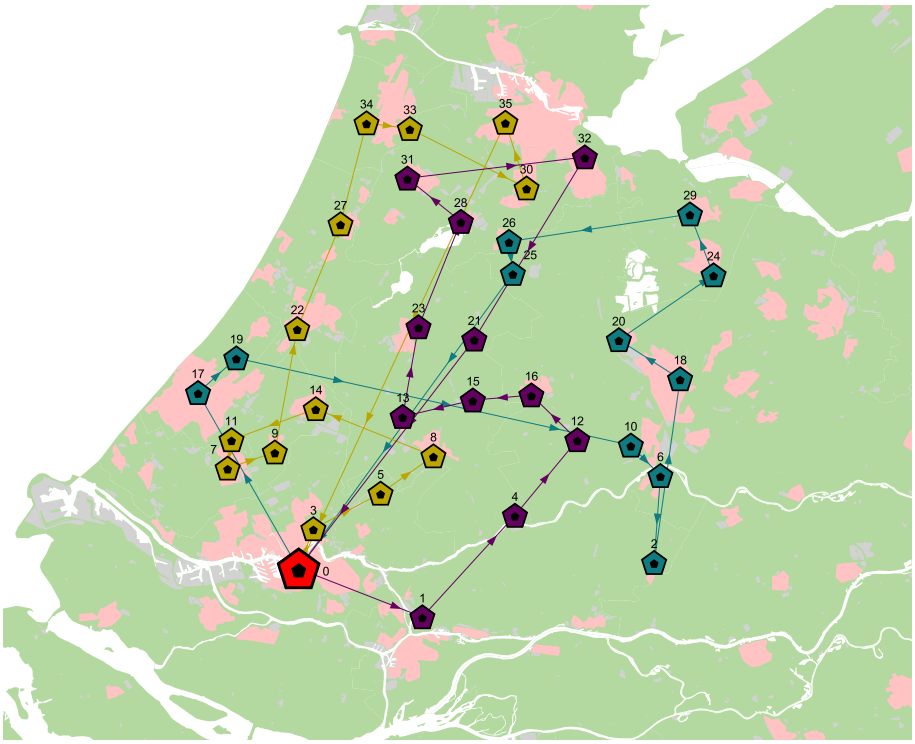

Figure C-7: The solution of the real-life problem taking five time-zones

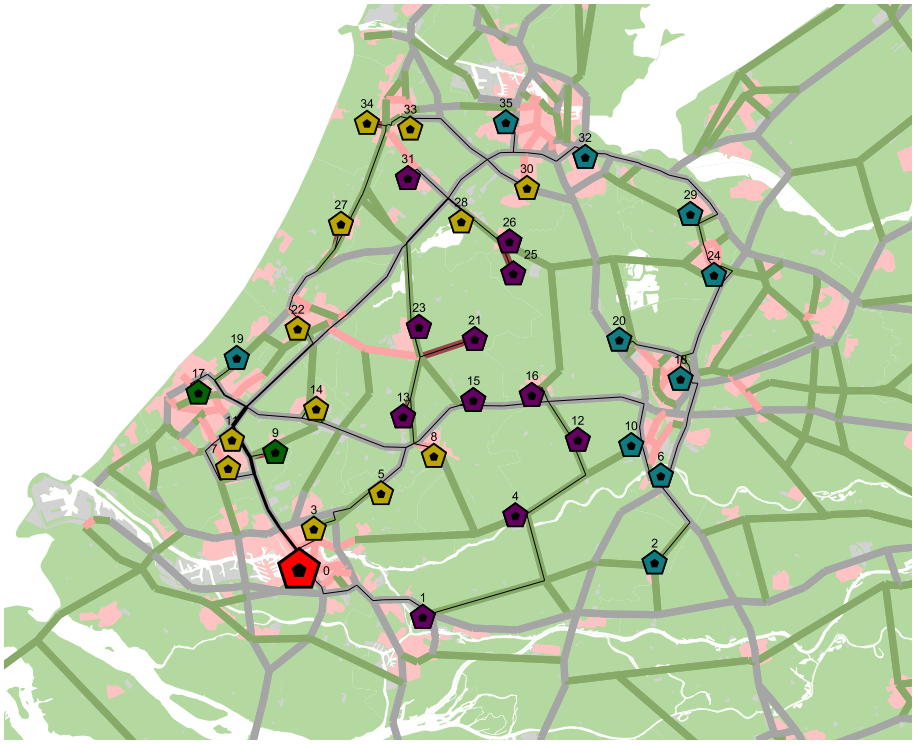

Figure C-8: The roads used of the real-life problem taking one time-zone

Figure C-9: The roads used of the real-life problem taking five time-zones

Bibliography

Aarts, E. and Lenstra, J. K. (1997). *Local Search in Combinatorial Optimization*. The Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, Chichester.

Baldacci, R., Battarra, M. and Vigo, D. (2008a). Routing a heterogeneous fleet of vehicles. In *The Vehicle Routing Problem: Latest Advances and New Challenges*, ed. by Golden, B., Raghavan, S. and Wasil, E. Operations Research/Computer Science Interfaces Series, Vol. 43. Springer.

Baldacci, R., Christofides, N. and Mingozzi, A. (2008b). An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming: Series A and B*, 115, 351–385.

Bräysy, O. and Gendreau, M. (2005). Vehicle routing problem with time windows, part ii metaheuristics. *Transportation Science*, 39(1), 119–139.

Dantzig, G. and Ramser, J. (1959). The truck dispatching problem. *Management Science*, 6(1), 80–91.

Dijkstra, E. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1, 269–271.

Donati, A. V., Montemanni, R., Casagrande, N., Rizzoli, A. E. and Gambardella, L. M. (2008). Time dependent vehicle routing problem with a multi ant colony system. *European Journal of Operational Research*, 185, 1174–1191.

Dorronsoro, B. (2007). VRP Web. `http://neo.lcc.uma.es/radi-aeb/WebVRP`, 2009.

Figliozzi, M. A. (2009). *A route improvement algorithm for the vehicle routing problem with time dependent travel times*. Paper under review for journal publication and also available in the proceedings of the 88th Transportation Research Board annual meeting, Washington DC, USA.

Fleischmann, B., Gietz, M. and Gnutzmann, S. (2004). Time-varying travel times in vehicle routing. *Transportation Science*, 38(2), 160–173.

Gambardella, L. M. (1999). MACS-VRPTW: A Multiple Ant Colony System for Vehicle Routing Problems with Time Windows - Numerical results for the Solomon problems. `http://www.idsia.ch/~luca/macs-vrptw/solutions/welcome.htm`, 2009.

Gendreau, M., Potvin, J.-Y., Bräysy, O., Hasle, G. and Lokketangen, A. (2008). Metaheuristics for the VRP and its extensions: a categorized bibliography. In *The Vehicle Routing Problem: Latest Advances and New Challenges*, ed. by Golden, B., Raghavan, S. and Wasil, E. Operations Research/Computer Science Interfaces Series, Vol. 43. Springer.

Haghani, A. and Jung, S. (2005). A dynamic vehicle routing problem with time-dependent travel times. *Computers & Operations Research*, 32(11), 2959–2986.

Hashimoto, H., Yagiura, M. and Ibaraki, T. (2008). An iterated local search algorithm for the time-dependent vehicle routing problem with time windows. *Discrete Optimization*, 5, 434–456.

Hasle, G. (2009). Transportation Optimization Portal of SINTEF Applied Mathematics - Problems and benchmarks - VRPTW - Solomon benchmark. `http://www.sintef.no/Projectweb/TOP/Problems/VRPTW/Solomon-benchmark`, 2009.

Homberger, J. and Gehring, H. (1999). Two evolutionary metaheuristics for the vehicle routing problem with time windows. *INFOR*.

Ichoua, S., Gendreau, M. and Potvin, J.-Y. (2003). Vehicle dispatching with time-dependent travel times. *European Journal of Operational Research*, 144, 379–396.

Malandraki, C. and Daskin, M. S. (1992). Time dependent vehicle routing problems: Formulations, properties and heuristic algorithms. *Transportation Science*, 26(3), 185–200.

van Rijswijck, H. (2008). *Dynamisch netwerk RESPONSE^{TM}*. TNO-Report. TNO Built Environment and Geosciences, Business Unit Mobility and Logistics, the Netherlands.

Soler, D., Albiach, J. and Martínez, E. (2009). A way to optimally solve a time-dependent vehicle routing problem with time windows. *Operations Research Letters*, 37, 37–42.

Solomon, M. (1983). Solomon's benchmark. `http://web.cba.neu.edu/~msolomon/problems.htm`, 2009.

van Staalduinen, J. (2009). Optimalisatie en ontwerp van logistieke netwerken. `http://www.tno.nl/response`, 2009.

Toth, P. and Vigo, D. (2001). *The Vehicle Routing Problem*. SIAM Monographs on Discrete Mathematics and Applications, Philadelphia, PA, USA.

Uil, K. (ca. 2004). *Response methodology*. Internal document. TNO Built Environment and Geosciences, Business Unit Mobility and Logistics, the Netherlands.

Verkeer en Waterstaat (2009). Kilometerprijs - wat gaat u betalen? `http://www.verkeerenwaterstaat.nl/onderwerpen/mobiliteit_en_bereikbaarheid`, 2009.

Verweij, K. (2009). TNO - Markets - Mobility - Products - Logistics and Supply Chain Management. `http://www.tno.nl/logistics`, 2009.

Vlaar, S. (2009). TNO - About Us - Organisation. `http://www.tno.nl/content.cfm?context=overtno&content=overtno&item_id=32`, 2009.

van Woensel, T., Kerbache, L., Peremans, H. and Vandaele, N. (2008). Vehicle routing with dynamic travel times: A queueing approach. *European Journal of Operational Research*, 186, 990–1007.

Index