# En-route rescheduling of home deliveries

*A case study on the home delivery operation of a large retail organization in the Netherlands.*

Peter Bijl
8-2-2016
University of Twente & Simacan B.V.

# En-route rescheduling of home deliveries

*A case study on the home delivery operation of a large retail organization in the Netherlands.*


| | |
|---|---|
| Author: | Ir. P.M. Bijl |
| Supervisors University of Twente: | Dr. ir. M.R.K. Mes<br>Prof. dr. ir. E.C. van Berkum |
| Supervisor Simacan B.V.: | Ir. M.E.J. Loot |
| Date: | 8-2-2016 |

# Preface

This is my thesis “En-route rescheduling of home deliveries” to conclude the track Production and Logistic Management within the master program Industrial Engineering and Management at the University of Twente. I wrote my thesis at Simacan: an IT company that specializes in making traffic, logistic and geospatial data accessible and useful for primary business processes. Simacan is established in Amersfoort.

During my graduation project, I encountered moments that made my research sometimes progress slower than I hoped. I want to thank my girlfriend and many of my friends for the moral support. Difficult moments occurred when my own simulation models turned against me, but I am very proud of the results with which I can conclude my master program.

I really liked my time in Amersfoort and at Simacan. In fact, I liked it so much that I will continue living in Amersfoort, with my first job at Simacan. I also want to thank everyone at Simacan for their help, patience, and all of the espressos. Rob and Felix, thank you for the opportunity to do my graduation project at your company. We will continue to do beautiful and innovative projects in the traffic and logistic sector!

My special thanks goes out to my company supervisor Martijn Loot for his feedback, and all the times he helped me with some of the difficult topics in my thesis. My special thanks also goes out to my University supervisors Martijn Mes and Eric van Berkum for their feedback during our sessions on the progression of my research.

I hope you will enjoy reading my thesis. If you have any questions regarding the research, do not hesitate to contact me.

Peter Bijl
Amersfoort, February 2016

# Management summary

The Delivery Control Tower (DCT) is a software product of Simacan that is designed for monitoring and communication in home delivery operations of retailers. Since the recent introduction of this product at a large Dutch retailer, a team of six people uses the product to monitor the home delivery operation, communicating with the vehicle drivers as well. Real-time vehicle positions and expected arrival times are calculated for all customers on delivery routes in the DCT. In case of expected lateness of delivery, drivers call the control room. They determine en-route rescheduling solutions to reduce the number of late deliveries. They use three basic strategies for rescheduling: 1) intra-route switching in the sequence of customers, 2) inter-route helper actions, where a helper vehicle drives towards a disrupted vehicle to transfer orders, or 3) accepting the lateness of delivery. They use relatively simple rules to determine their solutions, only utilizing moving expected late deliveries forward in the sequence or using empty vehicles as helper vehicles.

The current rescheduling process is labor-intensive, and non-optimal rescheduling solutions are often proposed. Therefore, Simacan wishes to introduce an en-route rescheduling method that automatically triggers on lateness, suggesting the best rescheduling solution. They wish to implement this in a Decision Support System (DSS) for control room teams to receive automated suggestions for en-route rescheduling. Therefore, we define our research goal as follows:

*"To develop a methodology for en-route rescheduling of home deliveries"*

The subject of en-route rescheduling is scarce in literature. Few papers consider en-route rescheduling with time windows, and they mainly consider vehicle breakdowns and changed customer wishes as triggers.

### Rescheduling method

We propose to solve en-route rescheduling problems using a Tabu Search Rescheduling (TSR) method. We use Tabu Search as our metaheuristic to guide our local search problems of rescheduling because of the quality, flexibility, and speed in finding feasible solutions. Our TSR method consists of three phases: triggers, selection of a relevant helper vehicles, and the rescheduling calculations. We test triggers on expected lateness, vehicle defects and vehicle breakdowns. When we trigger for rescheduling, we determine relevant vehicles we want to evaluate for helper-actions based on helper distance and the number of deliveries to be made by the helper. We evaluate all helper vehicles by insertion of the transfer location using our adapted version of the Solomon I1 insertion heuristic, where we include waiting times. We test 2-opt*-, or-opt- and exchange-operators for inter-route Tabu Search to find the best inter-route solution given a transfer location we select. Simultaneously with the evaluation of inter-route transfers, we use Tabu Search to find the best intra-route switching solution.

We determine the solution quality of the initial solution, all possible helper actions and the intra-route switching solutions using an objective function. We select the best solution found to continue the routes of the en-route vehicles using a function with the total time needed for the vehicles, time window violation penalties, and an exponential robustness bonus function. We translate all variables into time to get a single result from our function that we use to select the rescheduling solution.

### Simulation model

To test the TSR method performance, we develop a simulation model. This model aims to find good TSR method settings, while showing the potential savings of our method as well. We run the simulation model on two weeks of planning instances of the case study retailer, which include morning (PO) and afternoon (PA) shifts. We verify our simulation model, and we find that we overestimate the number

of late deliveries. We conclude that we model reality close enough to make a fair comparison between experimental settings. The total time fits the real situation well and the numbers of late deliveries are comparable to the real situation. The model enables us to show the results of our TSR method well, but the precise benefits of the method might differ slightly when actually implemented in the DCT.

### Results

Our TSR method is able to reduce the number of late deliveries with 63% for new customers and 24% for recurring customers in our simulation model. No more manual work by the control room is involved because we automatically trigger and calculate rescheduling solutions that can be implemented in a decision support system. We find that the TSR method works best with TSR triggers (without robustness trigger), a linear penalty function for time window violations, an exponential bonus function for more robust solutions, customer distinction between recurring and new customers, and the use of only or-opt-operators in our method. Figure 1 shows the simulation results over all shifts in the simulation data, where we are able to reduce the late deliveries for all shifts in the test set.

In additional analyses, we show that that on time departure can reduce the number and severity of late deliveries. We show that the effect of knowledge on future travel times is large, meaning that a good travel time prediction is key in en-route rescheduling. Finally, we find that our reduction of late deliveries is mostly caused by intra-route switching.

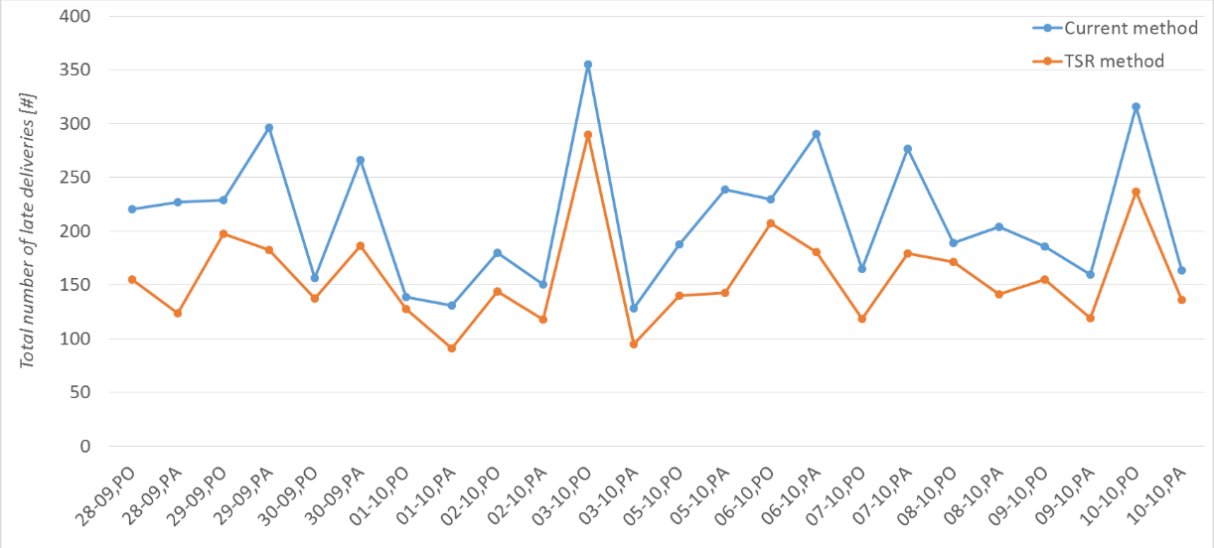


*Figure 1: total number of late deliveries (sum of new and recurring customers) for every day in the simulation comparing the current method and the best experimental settings for the TSR method.*

### Recommendations

We recommend the implementation of the TSR method as a DSS in the DCT. We prove the effectiveness of the TSR method in our simulation and find good settings for implementation. As a first version, Simacan can consider to only implement the intra-route switching part of our TSR algorithm. This can ensure a large decrease in late deliveries already, while being easier to implement in a live software product. The inter-route transfers are mostly lucrative and relevant in extreme cases. Eventually it is interesting to automatically support decision making for transfers as well.

The TSR method is suitable for more general application, so it can be used for en-route rescheduling problems with time windows in general. Although the algorithm is demonstrated on a retail home delivery case, more general applications can be found in other sectors such as parcel delivery services.

# CONTENTS

# 1 INTRODUCTION

This document contains the master thesis of Peter Bijl. The subject of the thesis is en-route rescheduling of home deliveries. We develop a method and test it on a case study of a home delivery operation of a large retail organization in the Netherlands.

Section 1.1 introduces Simacan B.V. (initiator of the research project) and background information on the research problem. The motivation of the project is further described in Section 1.2, focusing on the added value of our research from a literature point of view. We describe the research goal, research questions and research scope subsequently in Section 1.3, Section 1.4 and Section 1.5. Finally, we set out the research approach we use to achieve our research goal in Section 1.6.

## 1.1 CONTEXT

This master thesis is written in cooperation with Simacan B.V., a company in software products and that integrates real-time traffic information in primary business processes of other companies. They specialize in making geospatial data accessible and useful for both the traffic and the logistics sector. An important product of Simacan is a home delivery specific product called the Delivery Control Tower (DCT). The Delivery Control Tower (DCT) is a product of Simacan that provides overview and control in home delivery operations.

The DCT calculates Estimated Times of Arrival (ETA's) for all vehicles in the home delivery operations for all positions that they communicate to the DCT (generally one update per minute). The monitoring process in the DCT uses real-time insight in vehicle positions and traffic information. The traffic information is based on TomTom traffic data. Simacan uses an arrival time prediction framework in the Delivery Control Tower for ETA calculation in multi-stop delivery routes. Stops are made for all customers in the case of home deliveries. The Simacan ETA framework in the DCT contains the following modules:

- Next-Stop (NS) travel time module: predicts the driving time of a vehicle towards the next three stops of a trip. This part of the ETA prediction is based on instantaneous travel times[1] using real-time travel time information of TomTom.
- Service time module: predicts the service time needed for a stop. This is based on the planned service time that is determined by the retailer with the home delivery service.
- Between-Future-Stops (BFS) travel time module: predicts the driving time between stops that are both after the next three stops in the delivery trip. This part of the ETA prediction is based on a trajectory travel time[2] method.
- Break prediction module: predicts the moment that a vehicle driver will take a break. Time and duration of breaks are based on the retailer specific driving time rules. Planned breaks are used as predicted breaks in the current implementation within the DCT.

[1] Instantaneous travel times: The instantaneous travel time is defined as the travel time of a hypothetical vehicle traveling through the considered section at a speed profile identical to that of the present local speeds at all road segments (Treiber & Kesting, 2013).

[2] Trajectory travel times: determining a route travel time by using expected travel times for all road segments on the route that are based on road-segment-, day-of-the-week-, 5-minute-specific profiles of TomTom. The expected start time for a road segment is based on the expected time needed for all previous road segments on the route.

The case study used in this research is the home delivery operation of a large Dutch retailer. The retailer makes about 7,000 home deliveries on a daily basis. They do this with over 250 vehicles and two daily delivery shifts; morning (PO) and afternoon (PA). Throughout the Netherlands, they use 14 different distribution locations where vehicles depart. All trips made by the vehicles are monitored and controlled by a control room team. The use the DCT monitoring for this. Six people work during both shifts to supervise the home delivery operation.

Figure 2 displays an overview in the DCT that is used for monitoring by the control room team. The screenshot shown is of deliveries in region Utrecht. The following information is shown in this figure:

1. Current vehicle positions;
2. Name of the trip made by one of the vehicles;
3. Name of the distribution location where the vehicle starts its trip;
4. Current deviation of a trip compared to the planned arrival times;
5. Summary of current deviations of the selection (based on e.g., location, trip name) that are (or will be) late as fraction of the total number of customers in the selection.

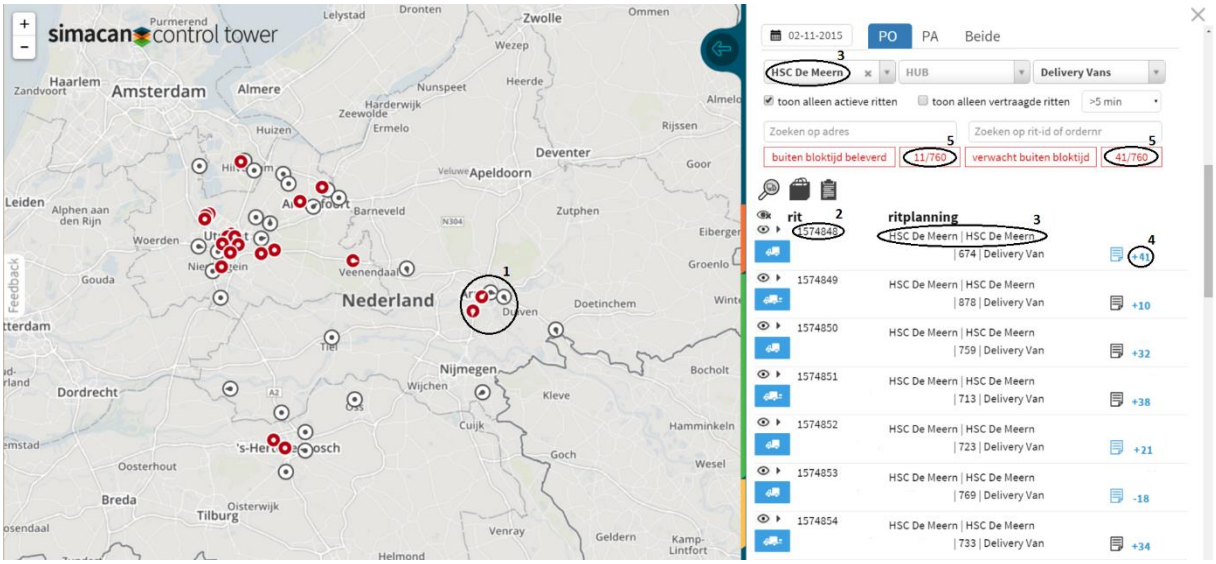


*Figure 2: screenshot of an overview of vehicles driving in region Utrecht within the DCT.*

When a customer places an order for home delivery, they select a time window for their order. This time window can range from one to six hours, where more narrow time windows often correspond to higher delivery cost. The retailer aims to deliver all products within the specified time windows to keep the service quality as high as possible. To do this, they sometimes use rescheduling of vehicles when problems occur. Rescheduling means that they alter an original plan to be able to make more deliveries in time. The rescheduling is en-route when all vehicles involved are already driving their delivery routes.

Both the control room team and the vehicle drivers observe possible problems regarding late deliveries. The control room team and the vehicle driver call each other and the control room team determines solutions. The control room team has the following options for rescheduling to solve expected late deliveries:

1. Switch: several customers receive products from one vehicle, and the sequence of these customers is planned beforehand. It is possible to deliver the customers in a different sequence when the time windows allow this. Additional distance might be travelled by the

vehicle, but this is accepted when the switch ensures that all deliveries are within the customer time windows.

2. Transfer: when a rescheduling involves one vehicle helping another by taking over some customers in order to be in time, this involves a transfer of the ordered goods. In this case, one of the vehicles is 'disrupted', and the other vehicle is a 'helper'. The control room team determines the customer orders to be transferred and a meeting location. Although the control room team is allowed to make transfers between two full vehicles, generally only transfers are made with on empty vehicle to minimize the risk of lateness later in the trip.
3. Accept & inform: when no possibility to ensure on-time delivery is found by the control room team, the customer service department of the retailer is informed. The customer service informs the customer about the lateness of the delivery.

When making rescheduling decisions, trip- and customer specific information is used. The trip specific information can be shown using the little arrow as displayed in Figure 2. Figure 3 shows an example of this information. The trip, with all customers that have to be visited, is displayed on the map. The following information is available in the list:

1. The planned start and end of the service time at a customer;
2. The realized start and end of the service time at a customer: a green edge means that the delivery was in time, a red edge means that the delivery was more than 20 minutes later than planned, and a red rectangle means a delivery outside the time window;
3. A timeline of the delivery: the blue part is the time window of the customer, the black part is the planned service time, the green/red triangle shows a realized time, and the grey rectangle shows the current time;
4. The number of goods in the customer order;
5. Expected deviation of arrival time from the planned arrival time based on current lateness and ETA-information.

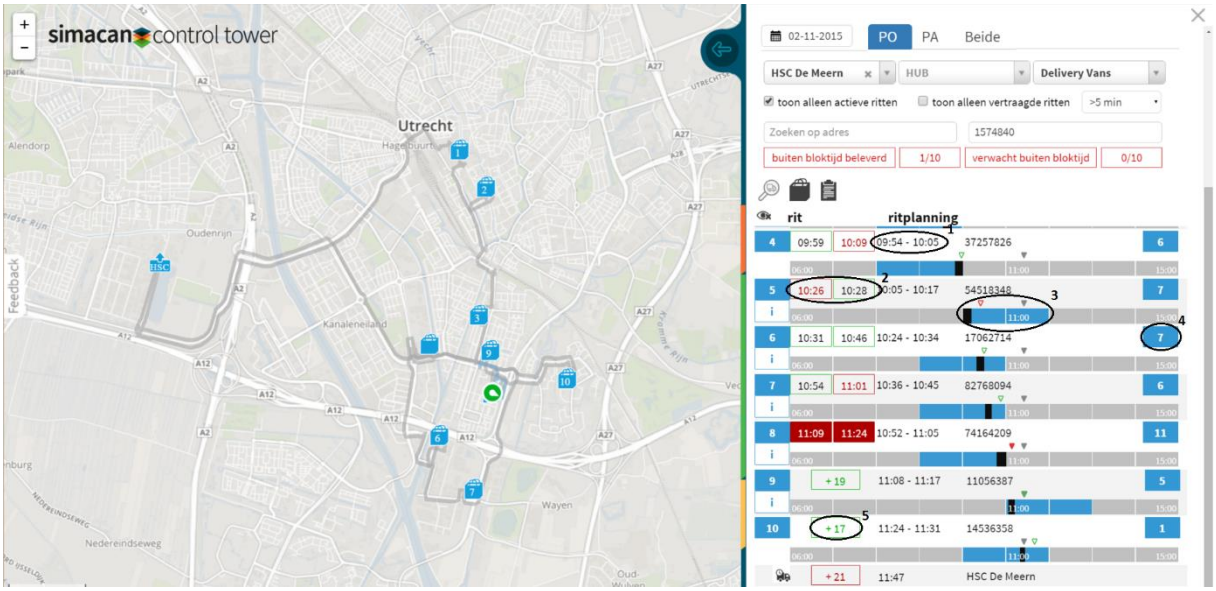


*Figure 3: screenshot that shows trip-specific information within the DCT.*

When asked to give an estimation on the number of rescheduling operations, the Dutch retailer from our case study stated that about ten percent of all trips are rescheduled. Most of these rescheduling operations only involve the switching of customers within a trip.

The current rescheduling leaves much room for human errors. The control room team is busy and has to determine good solutions within seconds. Simacan wishes to research an additional feature that can be part of the DCT. They want to introduce a Decision Support System (DSS) for the en-route rescheduling as described based on the current traffic situation on the road, specifically for home delivery operations. The retailer of our case study has a strong focus on service quality. The DSS should enable the control room team to make better en-route rescheduling decisions for home deliveries. Furthermore, the rescheduling solution should be robust for further changes, meaning that rescheduling does not occur too often or too easily.

We design an algorithm to be used in the DSS for en-route rescheduling of home deliveries in this study. Because are not able to test the rescheduling of home deliveries in real-life because of cost, time and control on the operations, we set up a simulation that enables us to predict system performance and the effects of our method on system performance.

## 1.2 MOTIVATION

As mentioned, the current rescheduling process by control room employees leaves room for human errors, while being labor-intensive at the same time. The people that determine the en-route rescheduling are busy and they have to determine good solutions within seconds. The practical motivation of our study comes from this point of view, helping control rooms to make better en-route rescheduling decisions.

For the motivation from literature, we first need to mention that home deliveries are planned on expected travel times between different customer pairs without any knowledge of the actual traffic situation at the time of travel. Expected service times for all customers are planned based on customer characteristics. The planning of vehicle routes before departure of the vehicles is a well-known NP-hard combinatorial optimization problem (Vehicle Routing Problem or VRP) that has received considerable attention in the scientific community.

We define routes and paths here because this study is influenced by both vehicle routing research and traffic engineering research. Different definitions for 'routes' are used in both research areas, ensuring possible confusion when using this term. We use routes in the vehicle routing research meaning, so this is defined as a sequence of stops for a vehicle. A path is defined as a list of road segments that is used to drive from one stop to another, so the actual path used by the vehicle to drive from point A to point B.

During the execution of the planned home deliveries, unexpected events can occur, causing travel times or service times to be either shorter or longer. In case of unexpected delays in the delivery process, conditions that are set by the home delivery company might no longer apply for the given operation. Examples of unexpected events in a home delivery process are: severe traffic conditions, accidents on the road, and the breakdown of vehicles. The events can ensure the need for deviation from the original schedule. A problem that consists of defining a new schedule for a set of en-route trips emerges. This problem is often referred to as the Vehicle ReScheduling Problem (VRSP). Other names for similar problems might be found in literature as well. The VRSP is essentially a class of VRP problems where extra constraints apply.

VRSP research is scarce in literature (Visentini, Borenstein, Li, & Mirchandani, 2014), but some examples are known. Li, Mirchandani, and Borenstein (2007) approach the VRSP as a dynamic version of the classic VRP. VRSP research has often focused on disruptions based on vehicle breakdowns (Li, Borenstein, & Mirchandani, 2007; Li, Mirchandani, et al., 2007; Li, Mirchandani, & Borenstein, 2009). The main objective of the VRSP is to minimize the involved operation and delay costs, under the

condition that uncompleted trips, including the disrupted one, must be finished (Li, Borenstein, et al., 2007; Spliet, Gabor, & Dekker, 2014). Ferrucci and Bock (2014) researched a very similar case to our research problem under the name of the Dynamic Pickup and Delivery Problem with Real-Time Control (DPDPRC). They consider minimizing lateness as the primary objective, and minimizing vehicle operating cost as a secondary objective. They consider dynamic events (e.g., traffic congestion) and orders that arrive dynamically over time. Because pick-up is also part of their problem, transfers between vehicles are prohibited in their methodology (except in case of vehicle breakdowns).

The future work on the dynamic problems mentioned by Ferrucci and Bock (2014) is, among others, to reduce vulnerability of a plan caused by unexpected dynamic events. Methods for improving the robustness of generated plans should be explored and integrated. They also state that interesting possibilities can be found in the anticipation of traffic congestion and time-dependent differences in travel times. The review work on the VRSP by Visentini et al. (2014) states that disruptions and real-time recovery for buses and trucks are topics that need to be further explored. They state that robustness in rescheduling is interesting to introduce when evaluating rescheduling options.

This research project distinguishes itself from other research by using a combination of triggers (the moments when rescheduling is suggested), transfers of goods between vehicles, and an en-route rescheduling method that reschedules home deliveries with time windows. In this context, we also introduce robustness in an objective function for the rescheduling in order to get more future-problem-proof new plans. The development of a fast heuristic solution approach for an en-route vehicle routing problem in our research project is an important contribution to the practice of rescheduling within the DCT, and a scarce subject within literature. The wish of Simacan to set up a general solution approach for the problem ensures that the methods developed here will be used by multiple home delivery operations or postal services in the future, adding even more practical value to our research project.

## 1.3 RESEARCH GOAL

The aim of this research project is to achieve the following research goal:

*"To develop a methodology for en-route rescheduling of home deliveries."*

## 1.4 RESEARCH QUESTIONS

The following research questions will be answered in the research process to finally achieve the research goal:

1) What is the current state of research on en-route rescheduling related subjects?
2) What is the current rescheduling process in the home delivery case study?
3) How can we ensure robust en-route rescheduling of home deliveries?
    3.1) When is it necessary to reschedule home delivery routes and what are relevant triggers to signal this?
    3.2) How can we determine selection of a search space in the en-route rescheduling process?
    3.3) What is a good method to reschedule customers within and between en-route vehicles?
4) How can we test rescheduling performance of the developed algorithm?
5) What performance can we expect from the methods developed in 3)?
    5.1) What are good settings for our rescheduling algorithm?
    5.2) Which managerial insights can be obtained regarding the en-route rescheduling algorithm?

## 1.5 RESEARCH SCOPE

This section specifies the scope of our research project. The moment of re-optimization takes place when all vehicles are loaded and driving towards their customers. This is of big influence on constraints and possible operational decisions.

The geographic scope of our research project is the Netherlands. Algorithms can also be applied in other countries, but our case study is limited to a home delivery operation that does home deliveries within the Netherlands. A set of data is available for the period September 7$^{th}$ to October 11$^{th}$. This will ensure that the research project includes five weeks of both morning and afternoon shifts. More data cannot be collected since the operation has not been completely monitored before September 7$^{th}$. End date is set so October 11$^{th}$ because of the total project lead time and data privacy issues of the retailer.

In this research project, no complete information about the real-time actions and results of rescheduling by the control room team is available. Since it is important to know the benefits of our algorithm, benchmark data will be based on a simple heuristic that follows from decision rules used by the control room team. Note that team members sometimes also use other knowledge or rules, but that it is impossible to include all human behavior in a simple benchmark policy.

## 1.6 RESEARCH APPROACH

This section introduces the research approach that will be used to achieve the research goal described in Section 1.3. Figure 4 displays the research approach we use in this study. The first part of the project introduces the research project in general.

Chapter 2 describes the existing literature on vehicle routing problems and vehicle rescheduling problems. We end this chapter with a research gap that we find from the literature study. Chapter 3 describes the process & data analyses of our case study. Chapter 4 defines the algorithms we develop to solve the problem of en-route rescheduling. Chapter 5 describes the simulation setup we use for evaluation of our algorithms. Chapter 6 presents the results of the rescheduling method in the simulation setup. We end Chapter 3 to Chapter 6 with a chapter conclusion that we do not show in the research approach. Chapter 7 shows the conclusions & discussion that follow from our study.

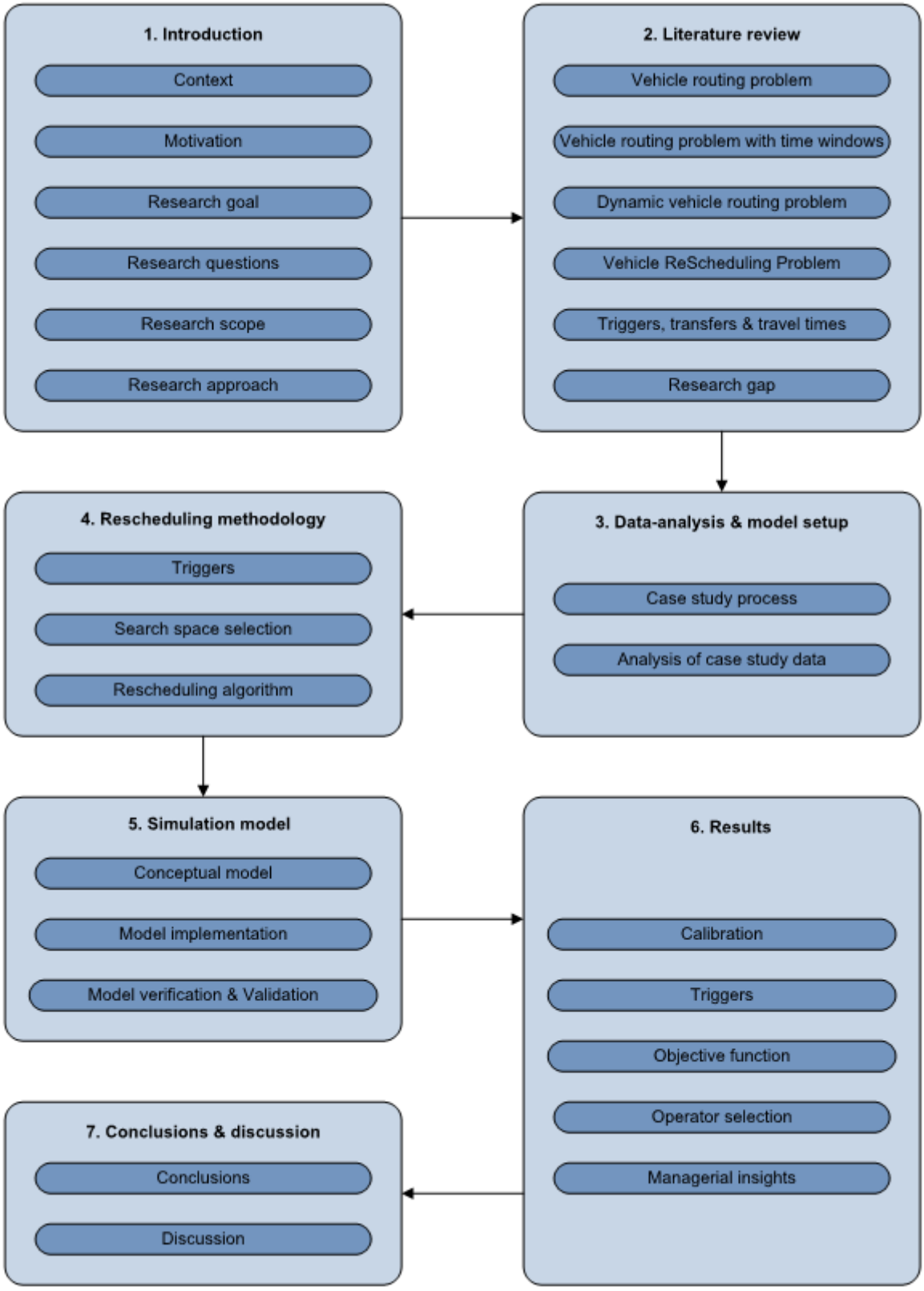


*Figure 4: Research approach for IEM master thesis project of Peter Bijl.*

# 2 LITERATURE REVIEW

This chapter reviews literature on relevant subjects for our research project. Aim of this chapter is to get an overview of methods for related problems to our research problem, and to find directions for good solution methods in our project.

Section 2.1 presents a brief introduction on the general Vehicle Routing Problem (VRP) and its many variants. Section 2.2 describes the VRP with Time Windows (VRPTW), which is very relevant in context of our research problem because we deal with a time window problem here. Section 2.3 and Section 2.4 describe literature on two relevant sub classes of the VRP: the Dynamic VRP (DVRP) and the Vehicle ReScheduling Problem (VRSP). Section 2.5 describes the availability of other research on relevant subjects for our research problem: triggers and transfers. Section 2.6 ends this chapter with a research gap and methodology conclusion from the discussed literature.

## 2.1 VEHICLE ROUTING PROBLEM

The Vehicle Routing Problem (VRP) is an optimization problem that aims to find a solution to service customers with a given fleet of vehicles. The objective is to find the route with minimal cost where all customers are served by the fleet of vehicles. Cost is generally expressed in distance or travel time. The VRP is an NP-hard combinatorial optimization problem that was first introduced by Dantzig and Ramser (1959) under the name of 'truck dispatching problem'. The VRP holds a central place in distribution management and it has become one of the most widely studied problems in combinatorial optimization (Cordeau, Gendreau, Laporte, Potvin, & Semet, 2002).

In VRP literature, a distinction between many different VRP variants is made. Some important variants of the VRP are the following:

- Open Vehicle Routing Problem (OVRP): same as the general VRP, only difference is that vehicles do not have to return to the depot.
- Capacitated Vehicle Routing Problem (CVRP): vehicles have limited capacity for the goods that have to be delivered.
- Vehicle Routing Problem with Time Windows (VRPTW): all service locations have time windows in which the service must occur.
- VRP with Pickup and Delivery (VRPPD): goods are picked up at specific locations, and they have to be delivered at other locations. This problem type is also referred to as the Pickup and Delivery Problem (PDP).
- Dynamic VRP (DVRP): the customers of the VRP that require service dynamically request service over time, ensuring that new plans need to be made dynamically as well.
- Vehicle ReScheduling Problem (VRSP): disruptions cause a planned schedule to be infeasible or too expensive. En-route modifications are made to the plan to ensure a feasible or less expensive plan.

Note that many other problem types are introduced by several authors. Some are very similar or even identical to others, but a new names are often introduced.

As mentioned in Chapter 1, we deal with time windows for the customers of this study. Therefore, we review literature on this subject first. After this, we review literature on the Dynamic Vehicle Routing Problem (DVRP). Next, we describe the Vehicle ReScheduling Problem (VRSP).

## 2.2 VEHICLE ROUTING PROBLEM WITH TIME WINDOWS

VRPTW literature makes a distinction in the use of hard and soft time windows. In case of hard time windows, no time window violations are allowed in the planning process. Extra vehicles are used if necessary. In the soft time window constraints research time window violations are penalized in the cost function, but they are allowed if this ensures a significant cost reduction for the total planning. We do not distinguish the two types in this chapter because solution methods are often similar. We describe exact algorithms for the VRPTW first. Next, we define criteria that we use to select a method in our research. After this, we describe heuristics and metaheuristics often used in VRPTW literature.

### 2.2.1 Exact algorithms for the VRPTW

An overview of exact algorithms is given by Cordeau, Laporte, Savelsbergh, and Vigo (2006). The methods they described are explained briefly below:

- Lagrangean relaxation: provides an approximate solution to the original problem with a simplification of the original problem by penalizing inequality constraints using a Lagrange multiplier.
- Column generation: this algorithm generates only the variables that have the potential to improve the objective function. It is based on the idea that only a subset of variables need to be considered when solving the optimization problem.
- Branch-and-cut: this algorithm uses a branch and bound algorithm while using cutting planes to tighten the linear programming relaxations.

Because the VRPTW is a complex and NP-hard combinatorial optimization problem to solve, real-time optimization of problem instances of the VRPTW is difficult. Solving VRP's exactly is time-consuming, even for small problem sizes (Cordeau et al., 2006). When a solution for a given situation in real-time is found, chances are that the situation already changed. Therefore, we focus on heuristic approaches in our problem instance. Heuristic approaches are more suitable in the context of our research project (Cordeau et al., 2002). Cordeau et al. (2002) state that heuristics for VRPTW's can be grouped in two categories: classic heuristics and metaheuristics. Both types of heuristics are described subsequently in the following sub sections.

### 2.2.2 Criteria for good VRPTW heuristics

Perspective on good heuristics is necessary to select our rescheduling heuristic algorithms. We use the four attributes of good VRPTW heuristics that Cordeau et al. (2002) define to select our rescheduling method. Cordeau et al. (2002) distinguish the following attributes for good VRPTW heuristics:

1. Accuracy: the degree of deviation of a heuristic solution value from the optimal value;
2. Speed: how fast the heuristic finds its heuristic solution for the problem;
3. Simplicity: measures how complicated or easy to understand the heuristic is;
4. Flexibility: how flexible the heuristic is to be applied to a large number of real-life applications.

Because we deal with real-time optimization, speed of our methods is even more important than for the regular VRPTW case. Exact optimization is in our case not possible due to problem size and available time.

### 2.2.3 Classic heuristics for the VRPTW

Classic heuristics are problem-dependent techniques that put an emphasis on quickly obtaining a feasible solution. A distinction within the classic heuristics can be made between construction and improvement heuristics. Additionally, some heuristics apply a postoptimization procedure on the result found.

#### *2.2.3.1 Construction heuristics*

Construction heuristics are applied when building an initial route for the VRPTW. The heuristics generally aim at obtaining a feasible solution in as little time as possible. Examples of construction heuristics are:

- Savings heuristic: the time window based saving heuristic as proposed by Solomon (1987) is an extension of the original savings based heuristic for VRP's by Clarke and Wright (1964). The heuristic starts with a separate route of depot – customer – depot for each customer. The savings of insertion of customer in a route are determined by the reduced cost when a customer is inserted into an existing route. In addition to taking into account vehicle capacity constraints, we must check time window constraints for violation at every step in the heuristic process.
- Nearest Neighbor (NN) heuristic: this heuristic starts every route by finding the non-routed customer "closest" (e.g., in distance or time) to the depot. At each iteration, the heuristic looks for the next customer "closest" to the last customer that was added to the route. This search is performed among all the customers that are feasible (on all constraints, so time windows, vehicle capacity, etc.) to add to the emerging route. A new route is started any time the search fails, unless there are no more customers to schedule (Solomon, 1987).
- Insertion heuristics: Solomon (1987) described three types of insertion heuristics that apply for the VRPTW. All types start with a tour of a subset of the set of vertices, extending it with additional vertices. A route is first initialized with a seed customer and the remaining non-routed customers are added into this route until it is full (looking at scheduling horizon and/or vehicle capacity). This is done until all customers are serviced. The seed customers are selected by finding either the geographically farthest non-routed customer in relation to the depot or the non-routed customer with the lowest starting time for service. The three types of insertion heuristics used are:
    - Insertion heuristic type I1 by Solomon is the most successful in evaluation (Bräysy & Gendreau, 2005a). This heuristic minimizes extra distance and extra time required to visit the customer that is evaluated for insertion.
    - The second type of insertion heuristics by Solomon I2 aims to select customers whose insertion costs minimize a measure of total route distance and time.
    - The third type of insertion heuristics by Solomon I3 accounts for the urgency of servicing a customer (Bräysy & Gendreau, 2005a).
- Sweep heuristic: the version of this heuristic without time window constraints was first proposed by Gillett and Miller (1974). It starts with a ray from the depot that sweeps clockwise (or count-clockwise), adding customers encountered. A new route is started when the vehicle is full. An example of the sweep heuristic with time windows is given by Solomon (1987). The VRPTW is split into subsequent clustering and scheduling stages. In the first stage, the original sweep heuristic (without considering time windows) is applied. Next, a center of gravity is computed and customers are partitioned according to angle. In the second stage, customers assigned to a vehicle are scheduled using an insertion heuristic.

#### *2.2.3.2 Improvement heuristics*

Improvement heuristics start with a feasible initial solution and then try to improve this by modification of this solution. Improvement heuristics are very relevant for our research problem, because an initial solution already exists at the moment of rescheduling.

To design an improvement heuristic, one typically needs to specify a move-generation mechanism that creates neighboring solutions by changing one attribute or a combination of attributes of a given

solution. Once a neighboring solution is identified, it is compared against the current best solution found. The current solution is used for further improvement (Bräysy & Gendreau, 2005a). Two types of moves (or improvements) apply for the VRPTW: intra-route improvements and inter-route improvements. Intra-route improvements are made within routes, inter-route improvements are made between routes.

First, we describe the possible neighborhood structures for the VRPTW. This is the basis of all research on both intra-route and inter-route improvement heuristics. Next, we describe examples of improvement heuristics.

**Neighborhood structures**

The basis of almost all route-improving heuristics is the notion of a neighborhood: the set of solutions that can be generated with a single modification of the current solution (El-Sherbeny, 2010). El-Sherbeny (2010) listed the following most used neighborhood heuristics:

- Relocate operator: the moving of one customer from one route to another;
- Exchange operator: the interchange of two customers;
- k-Opt operator: the interchange of one (or more) segment(s) of a route with one (or more) segment(s) within one route or between two routes;
- 2-Opt* operators: a case of the k-Opt operator that only allows inter-route interchange of 2 segments, while ensuring that no reversal of some portion of the route is realized;
- Or-Opt operator: the moving of a chain of one, two or three consecutive customers to a different location in search space. This type of operator is very suitable for time window problems;
- K-node interchange operator: each customer i, its successor j and the two customers closest to i and j on a different route are removed and the k most promising insertions of the four vertices are evaluated;
- λ-interchange operator: subset of customers of size ≤ λ is exchanged with a subset of customers of size ≤ λ from another route;
- Shift-sequence operator: the moving of a customer from one route to another while checking all possible insertion positions. If an insertion is feasible by removing another customer j, it is removed and inserted in another route.

Potvin and Rousseau (1995) describe operators specifically for the VRPTW. They state that the use of the 2-opt* operator and the Or-opt operator in heuristics that designed for time window problems yield good results. Both types of operators preserve order of customers when finding new solutions in the search process.

**Heuristics**

Intra-route optimization is often combined with inter-route optimization for the VRPTW. For instance, Lin (1965) tested 2-opt and 3-opt edge exchange procedures, both for intra-route and inter-route exchanges. He concluded that only less than 10% of the solution improvements involve the reversal of the orientation of a sequence of two or more customers. Other intra-route interchange heuristic approaches are researched by, for example, Calvo (2000) and Ohlmann and Thomas (2007).

Inter-route heuristics make adaptations to vehicle routes over the different routes that are already scheduled. In the context of improvement heuristics, we often start from a greedy or fast feasible solution. Changing the visits of customers can be done by the neighborhood structures as described. As mentioned, intra-route improvements and inter-route improvements are often combined.

Bräysy and Gendreau (2005a) presented a complete overview of search methods, comparing the speed and solution quality of the methods in six different classes. They use the original research Solomon (1987) for the 56 benchmark instances they use. These specific problems have 100 customers, a central depot, capacity constraints, time windows on the time of delivery, and a total route time constraint. The benchmark instances can be classified on location clustered (C) customers, location random (R) customers, and location mixed random and clustered (RC) customers.

Bräysy and Gendreau (2005a) make an additional distinction in their evaluation for short-haul (1) routes and long-haul (2) routes (where the short-haul routes generally require more vehicles). They evaluate the results on all individual benchmark instances and aggregate the results to the classes described to get feeling for solution quality of the algorithms in the different classes of instances. The best results on the benchmark instances are, in the comparison by Bräysy and Gendreau (2005a), found in the research of Caseau and Laburthe (1999), Schrimpf, Schneider, Stamm-Wilbrandt, and Dueck (2000), Cordone and Calvo (2001), and Bräysy (2002). The most of the methods use construction heuristics proposed by Solomon, and an improvement heuristic as described here:

- Caseau and Laburthe (1999) use swap, relocate, and flush and relocate as in their improvement phase. In swap, they remove a chain of consecutive customers on a route to another. In relocate, they remove a vertex from one route to another. This can result in relocating a vertex from that route to the next. The relocate move is followed by re-optimization of each route concerned by the move. In their last move, flush and relocate, Caseau and Laburthe (1999) remove all nodes that can be directly relocated into another route from a given route, before trying to insert a given customer (Bräysy & Gendreau, 2005a).
- Schrimpf et al. (2000) introduced a heuristic that the authors name 'ruin and recreate'. Three strategies are used to remove a set of customers from a solution: randomly, based on the distance to a randomly selected key customer and a set of succeeding customers on the same route with the key customer. The removed customers are then reinserted in random order using a greedy cheapest insertion heuristic.
- Cordone and Calvo (2001) proposed a deterministic heuristic based on classic k-opt exchanges combined with a procedure to reduce the number of routes. The special feature of their algorithm is that it alternates between minimization of total distance and of total route duration to escape from local minima (Bräysy & Gendreau, 2005a).
- A heuristic, referred to as B3 by Bräysy (2002), used multiple feasible starting routes, reducing the number of routes using a new ejection chain-based approach that also considers reordering of the routes. After this, Or-opt exchanges are used to minimize total traveled distance.

Some other interesting research is found in the work of Gendreau, Hertz, and Laporte (1992), who apply GENI and GENIUS methods. The GENI (GENeralized Insertion) method is a hybrid of tour construction and local search. Cities are inserted one by one and 3- and 4-Opt moves are performed after every insertion. The GENIUS (GENeralized Insertion Unstringing and Stringing) method by Gendreau et al. (1992) starts with a GENI solution, looks for the best deletion, followed by 3- and 4-Opt moves, and applies the GENI method to reinsert the deleted city in a route.

#### *2.2.3.3 Postoptimization procedures*

A postoptimization procedure is a heuristic that applies some extra simple modifications of the solution found to possibly minimize the cost even further. It is different from an improvement heuristic in the sense that it only modifies other variables than the route itself (e.g., departure time). We mention two examples of postoptimization procedures.

- Forward Time Slack was first introduced by Savelsbergh (1992). He introduced the concept of forward time slack that may be used to postpone the beginning of service at a given node without causing any time window violation. When feasibility must be maintained through exchanges, the forward time slack is the largest increase in the beginning of service at a customer that will not cause any time window violation (Cordeau, Laporte, & Mercier, 2004).
- Rochat and Taillard (1995) propose a postoptimization procedure by solving a set-partitioning problem. They determine the best solution using tours in a set of total solutions using local search. Their postoptimization procedure allows the solutions to be improved with a modest increase in computation time (Rochat & Taillard, 1995).

### 2.2.4 Metaheuristics for the VRPTW

Metaheuristics are problem-independent techniques that do not take advantage of any specificity of the problem. The metaheuristics are mainly based on two principles: local search and population search (Cordeau et al., 2002). The most frequently used metaheuristics for VRPTW instances, according to research of Cordeau et al. (2002), Gendreau (2002) and El-Sherbeny (2010), are listed below. The same neighborhood heuristics listed for the classic improvement heuristics are also applicable to be used in context of metaheuristics.

- Simulated Annealing (SA): a method that searches the neighborhood in a defined order, first introduced by Kirkpatrick (1984). In SA, neighborhood solutions are accepted as the new incumbent solution if either the solution is better, or the solution is worse but still accepted based on a certain probability. This temperature variable is lowered gradually in the search process, steadily reducing the probability that a non-improving solution will be accepted. The simulated annealing algorithm stops when a fixed consecutive number of searches fail to produce a new incumbent solution, or when a defined stop temperature is reached (El-Sherbeny, 2010).
- Tabu Search (TS): this technique was first introduced by Glover (1986). TS explores the search space by moving to the best solution in the direct neighborhood at each iteration. To avoid cycling, recently explored solutions or moves are declared forbidden using a Tabu list. When a solution or move is on this Tabu list, the move is not allowed and the next best move is selected (if not on the Tabu list as well). The duration that an attribute remains Tabu is called its Tabu tenure. TS is often terminated when no improvement is found for a given time or a given number of iterations.
- Genetic Algorithms (GAs): the concept of GAs was first introduced by Holland (1975). GAs are metaheuristics to find good solutions through the process of simulated natural selection. It basically consists of four phases: representation, selection, recombination, and mutation. The representation of the solution space consists of encoding significant features of a solution as a chromosome, defining an individual member of a population. Selection consists of randomly choosing two parents, where 'fit' parents (good solutions) are more likely to be picked. Recombination makes use of genes of the parents to generate offspring. Finally, mutation makes it possible that genes of the children as determined by the other steps are modified randomly (Bräysy & Gendreau, 2005b).
- Other promising examples of algorithms used in context of the VRPTW are:
    - Adaptive Memory Procedures (AMPs): the first AMP in context of the VRP was introduced by Rochat and Taillard (1995). The adaptive memory is a pool of routes taken from the best solutions visited during the search. Goal is to provide new starting solutions for local search methods through selection and combination of routes extracted from the memory. The selection of routes from the memory is done

probabilistically and the probability of selecting a particular route depends on the value of the solution to which the route belongs (Bräysy & Gendreau, 2005b).

- Variable Neighborhood Search (VNS): VNS is a metaheuristic that systematically changes the neighborhood structure, combining this with local search to get to local optima. First implementation of the heuristic in context of the VRPTW is found in Mladenović and Hansen (1997).
- Large Neighborhood Search (LNS): LNS is a metaheuristic where an initial solution is gradually improved by alternately destroying and repairing the solution. An important parameter is the size of the part of the solution that is destroyed. If this is only a small part of the solution, LNS acts as a regular local search method. If the part of the solution that is destroyed is large, LNS rebuilds new solutions from scratch. First implementation for the VRPTW is found in Shaw (1998).
- Ant Colony Optimization (ACO): The ACO is inspired by an analogy with real ant colonies looking for food. The ants mark paths they travel by a pheromone trail. This pheromone provides information to other ants that are attracted to it. With time, paths leading to the more interesting food sources are visited more frequently and theses paths are marked with larger amounts of pheromone (Bräysy & Gendreau, 2005b). Known implementations of this algorithm in the context of the VRPTW are provided by Gambardella, Taillard, and Agazzi (1999) and (Dorigo, Caro, & Gambardella, 1999).

Gendreau (2002) and Cordeau et al. (2002) both state that Tabu Search is the most suitable metaheuristic for the VRPTW as is finds good solutions fast. Gendreau (2002) states that Tabu Search methods generally perform better compared to the other metaheuristics. Furthermore, he stated that speed is a paramount criterion in real-time problems. Methods such as SA and GA slowly converge to good solutions, and are therefore less suitable in a real-time context. El-Sherbeny (2010) states, after an elaborate literature review of metaheuristics for the VRPTW, that applications show that Tabu Search is flexible and efficient. Furthermore, they found that Tabu Search gives results which compete or exceed those of the best heuristics known for many problem instances. Pillac, Gendreau, Guéret, and Medaglia (2013) state that Tabu Search particularly led to impressive results on different dynamic vehicle routing problems.

We outlined four attributes to select a good heuristic earlier. When evaluating the simplicity of the method, SA and Tabu Search would score about the same. GA is harder to understand however, thus scoring lower on the simplicity criterion. Cordeau et al. (2002) also scored a Taburoute implementation by Gendreau, Hertz, and Laporte (1994) with high accuracy, medium speed, medium simplicity, and high flexibility in their paper.

## 2.3 Dynamic Vehicle Routing Problem

In the Dynamic Vehicle Routing Problem (DVRP) requests for service arrive dynamically over time. The word dynamic indicates that information is revealed during the design or execution of routes. This class of problems is also referred to as online or real time (Jaillet & Wagner, 2008). Dynamic vehicle routing is attracting a growing attention from transportation companies (Respen, Zufferey, & Potvin, 2014). In recent years, much research in the DVRP literature focused on uncertain travel times, uncertain (or stochastic) demands and online requests (Bertsimas & Simchi-Levi, 1996; Gendreau & Potvin, 1998; Pillac et al., 2013).

Ichoua, Gendreau, and Potvin (2000) consider a long-haul courier service application. Their objective is to minimize a weighted sum of travel distance and lateness at customer locations. They enable

diversion by applying a variant of a real-time routing approach. Computational results show that diversion can significantly improve the solution quality.

Klundert and Wormer (2010) study a real-time problem of assigning servicemen to requests that arrive dynamically over time. They aim to optimize responsiveness, i.e., minimize the waiting in excess of a promised response time, and they do this in context of a practical use case of the ANWB (a Dutch roadside service association). They show that, when aiming to minimize end-of-day lateness, real-time assignment on a combination of travel distance and waiting time provides significantly better results than alternatives that directly rely on lateness objectives.

An interesting series of work on the vehicle routing problem with urgent delivery of goods is performed by Ferrucci and Bock (Ferrucci & Bock, 2014, 2015; Ferrucci, Bock, & Gendreau, 2013). In Ferrucci et al. (2013), they propose a new pro-active real-time control approach that exploits stochastic knowledge in order to actively guide vehicles into future request-likely areas in a use case where new customer demands are introduced dynamically. They introduce an objective function that determines the following cost per customer: *weight of customer * customer inconvenience + diversion penalty*. Here, customer inconvenience is operationalized by a function of request response times.

Ferrucci and Bock (2014) introduce the Dynamic Pickup and Delivery Problem with Real-Time Control (DPDPRTC) for urgent real-world transportation services. They consider minimizing lateness as the primary objective, and minimizing vehicle operating cost as a secondary objective. By optimizing consecutive static problem instances, which are snapshots of the ongoing transportation service, the authors buffer all dynamic events that occur during a set time. They consider dynamic events (or triggers) such as traffic congestion, vehicle slowdowns and vehicle breakdowns. All these dynamic events are considered in the next static problem instance.

In order to efficiently adapt the existing transportation plan to the consequences of dynamic events, Ferrucci and Bock (2014) apply a staged Tabu Search approach. Within the staged Tabu Search they consider: 1) within tour insertion, 2) relocation, 3) multi-stop relocation, 4) large neighborhood search, and 5) exchange of stops between tours. When no improvement is found for *X* iterations in one of the stages, the next stage is selected for further improvement.

It is important to note that the work of Ferrucci and Bock (2014) does not consider transfers when the relocation operator is used. They assume that transfers are not allowed, except if a vehicle breaks down. Pick-up is also part of their problem, meaning that the products are not yet loaded into one of the vehicles.

In another article Ferrucci and Bock (2015) focus on DVRP's in which new requests arrive over the day in a daily distribution process of urgent goods with the same objective function as before. The results they found are interesting for further research in VRPTW instances as well. Ferrucci and Bock (2015) state that it would be interesting to test the efficiency of the approach on routing problems with a different objective function as well, e.g., the minimization of travel time, or a combination of minimization of travel time and customer inconvenience.

## 2.4 VEHICLE RESCHEDULING PROBLEM

A review of the real-time vehicle schedule recovery problem is given by Visentini et al. (2014). In their review they distinguished three areas where the problems might emerge: road-based services (the VRSP), train-based scheduling and airline schedule recovery problems. We focus on literature on the road-based services here, since our use case is in this area specifically.

Li, Borenstein, et al. (2007) stated that the main objective of the VRSP is to minimize operation and delay costs, while serving the passengers or cargo on the disrupted trip and completing all remaining trips including the disrupted one. They developed a Decision Support System (DSS) for single-depot rescheduling to help human schedulers to find optimal schedules with and without unexpected incidents. They apply a procedure 'Build-Feasible-Networks' to build the set of all possible feasible networks and the forward-backward combined auction algorithm to find the minimal cost. Li, Borenstein, et al. (2007) found that their DSS can be an effective and efficient tool for real-time operational planning in transportation/logistic companies.

In another research project, Li, Mirchandani, et al. (2007) formulated a model for the VRSP and develop several fast algorithms to solve it with parallel synchronous auction algorithms. Computational results on randomly generated problems show that, for small problems, all of the developed algorithms demonstrate very good computational performances. For large problems, parallel auction algorithms (ensuring parallel computing) provide the optimal solution with small computation times.

In a different paper by the same authors, Li et al. (2009), introduced the real-time rescheduling of the VRPTW. An important issue with the time windows is related to route disruptions. If only operating costs and service cancellation costs are minimized, the original routes might be considerably disrupted. Thus, it is crucial to reduce the number of changes in the development of new routes since truck drivers may not be familiar with new delivery or pickup points. The strategy of Li et al. (2009) is to impose penalties on each route change in order to reduce route disruptions. They apply this in a Lagrangian relaxation that initially relaxes the hard constraints. The resulting Lagrangian relaxation problem is decomposed into constrained shortest path problems with time windows and vehicle capacity constraints. In order to obtain a solution for the Lagrangian relaxation problem quickly, a dynamic programming based heuristic solves the constrained shortest path problem heuristically and an insertion algorithm is used to obtain a feasible solution.

Mirchandani, Li, and Hickman (2010) researched the VRSP in context of bus breakdowns. They reassign and reschedule the bus fleet in case of a bus breakdown. They minimize the sum of operating costs, delay costs, schedule disruption costs, and trip cancellation costs. In their research, they combine the VRSP with bus signal priority, which can reduce bus delays at signalized intersections.

Mu, Fu, Lysgaard, and Eglese (2011) researched an instance of vehicle rescheduling for the capacitated VRP. They state that the VRSP is different from the VRP in five ways:

1) Vehicles do not depart from one depot, but they are at different locations when rescheduling occurs;
2) Computing time is a critical factor in the VRSP;
3) The solution to the original VRP can be used when solving the VRSP, while the VRP has to be solved from scratch;
4) Additional cost need to be taken into account for plan deviation when solving the VRSP, while the VRP only aims to minimize the cost given the relevant constraints;

5) Not all decisions are possible when solving the VRSP. In some cases, constraint violation might be unavoidable, and it will be more important to support the decision maker in its decisions. For the VRP, extra vehicles can be dispatched if the plan is not yet feasible.

In their research, Mu et al. (2011) found that a flexible and simple Tabu Search implementation, with a relocation neighborhood structure, gives very promising results in their test cases of the capacitated VRP.

A second study on the VRSP with time windows is found in Wang, Ruan, and Shi (2012). They researched disruption events in logistics delivery, focusing on changed customer wishes. The triggers are different from our research project. The nested partitioning they use to find a solutions for the VRSPTW performs well compared to other heuristics.

Another research project on the VRSP without the use of time windows is performed by Spliet et al. (2014). They experienced that Dutch retail companies currently reschedule manually when a long-term schedule no longer applies. They proposed a two-phase heuristic that is capable of finding good solutions within a small amount of computation time. Their numerical experiments show that solutions of the two-phase heuristic are on average close to optimal. The two-phase heuristic by Spliet et al. (2014) starts with an infeasible master schedule $S^M$ and modify it to make it feasible. The new feasible solution $S^{TP}$ is then introduced. In the first phase of the heuristic, a specific set of edges is removed from the master schedule. The second phase adds new edges to make the obtained schedule feasible, and ensuring that it has low deviation costs.

## 2.5 Triggers & transfers

This section introduces some additional relevant subjects for the en-route rescheduling of vehicles. First, we introduce triggers: situations that cause a need for rescheduling. Next, we describe transfers of goods between vehicles, which is relevant when inter-route modifications in the plan are made.

### 2.5.1 Triggers

In context of the vehicle rescheduling problem, certain triggers for rescheduling vehicles are already used. Triggers often considered in literature are vehicle breakdowns or accidents of the vehicle on a route (Li, Borenstein, et al., 2007; Mirchandani et al., 2010; Mu et al., 2011).

Another trigger for rescheduling might be found in the working hour regulations of truck drivers. In all member countries of the European Union and in many other countries, regulations for driving and working hours of persons involved in road transportation are effective (Kok, Meyer, Kopfer, & Schutten, 2010). It is therefore important to include the violation of these driver regulations as a trigger, since delays can ensure that drivers have to take long breaks to working hour regulations.

Wang et al. (2012) mention four types of triggers for the VRSP:

1) Vehicle disruptions: vehicle breakdowns or blocked vehicles;
2) Cargo disruptions: damaged cargo;
3) Customer disruptions: changed time windows, demand, or delivery addresses;
4) Combinational disruptions: a combination of multiple disruptions as mentioned in 1) to 3).

The most relevant triggers for our project are the vehicle disruptions, while Wang et al. (2012) only focus on the customer disruptions. Customer disruptions will not occur in our case because the plans are made when customer orders cannot be changed anymore.

Traffic delays can be an important trigger for rescheduling as well (Visentini et al., 2014). Huisman, Freling, and Wagelmans (2004) build in their research on this information for instances of the DVRP. However, they use the information before the actual departure of the vehicles from the depot. Still, traffic or service time delays are important to consider as triggers for rescheduling, provided that the delay is sufficiently large.

### 2.5.2 Transfers

When solving the VRSP as defined for our research project, transfers need to be included in the rescheduling since goods for customers are often already placed in the trucks. Respen et al. (2014) propose the assumption that the position of each vehicle is known at all times, thanks to accurate GPS devices. They show the positive impact of vehicle position tracking devices on solution quality. Their results indicate that a reassignment action, ensuring that a transfer will be necessary in our case, should be considered disruption of the current plan is detected.

Some literature on including transfers is found for the Pickup and Delivery Problem (PDP). Thangiah, Fergany, and Awan (2007) included transfers for shipments when these transfers lead to an improvement of the objective function in their research on the split – delivery pickup and delivery problem with time windows and transfers. If a vehicle arrives at a depot before a second vehicle, a shipment can be transferred to the second vehicle that visits this depot. Other research on the transfer of shipments is found in Bouros, Sacharidis, Dalamagas, and Sellis (2011). They distinguish two types of transfers, 'without detours' and 'with detours', and show their method includes both types of transfers in an objective function.

For people transfers, more research is available within the PDP. Cortés, Matamala, and Contardo (2010) added flexibility to the PDP formulation by providing the option for passengers to transfer from one vehicle to another at specific locations. To do this, they included special transfer nodes where multiple visits are allowed. Renaud Masson, Lehuédé, and Péton (2013) developed an efficient heuristic that inserts requests through transfer points with an adaptive large neighborhood search for the PDP in disabled people transport. This required the introduction of new destruction and repairing heuristics dedicated to the use of transfer points. In a follow-up research, Renaud Masson, Lehuédé, and Péton (2014) looked at the dial-a-ride problem with transfers, where requests of users between a set of pickup points and a set of delivery points occur in the presence of ride time constraints. They found that the introduction of transfer points led to non-negligible savings in their problem case using an adaptive large neighborhood search.

In their research on bus schedule recovery, Mirchandani et al. (2010) stated that the selection of the helper vehicle involves several factors such as the time when the trip was disrupted, the position of the currently operating buses, the available capacities of potential backup buses, and the compatibilities of itineraries among the bus trips.

Some more literature on the subject of transfers may be found in literature on different problem types. For example, intermodal transport and ride-sharing problems can also include transfers. However, we chose not to outline the literature on these subjects because of limited time for our review and because the different nature of the transfers in these other problem types.

## 2.6 LITERATURE REVIEW CONCLUSION

This chapter provides an overview of existing literature on relevant subjects for our research project. We describe the research gap and methodology conclusion from the literature review subsequently here. Because we use many overview papers (VRPTW: Cordeau et al. (2002), Bräysy and Gendreau

(2005b), El-Sherbeny (2010), DVRP: Pillac et al. (2013), VRSP: Visentini et al. (2014)) to review literature, we believe we present a complete overview of available literature on the relevant subjects.

### 2.6.1 Research gap

We find that detection and prediction of triggers is rare and quite new to the existing research. A lot of research is available for the VRPTW, but these methods are mostly suitable before the start of a home delivery operation. Still, many techniques for optimization that relate to the use of time windows are known in this domain, which is important for the context of our research.

We also provided an overview of research on the DVRP and VRSP. Generally speaking, the literature on vehicle rerouting for road transport is scarce (Spliet et al., 2014). Very few papers consider en-route rerouting with time windows, but some examples are available on the subject (Ferrucci & Bock, 2014; Li et al., 2009; Wang et al., 2012). These studies mainly consider vehicle breakdowns and change customer wishes however, which differs from our focus on lateness triggers in the home delivery operation.

The transfer of goods from one vehicle to another, as the case is in this research project, seems to primarily exist in the context of people transfer in (public) transport versions of the VRP. It has, to our knowledge, not yet been applied to the use case of en-route rescheduling of home deliveries.

We contribute to the current state of VRSP research by researching en-route rescheduling in home delivery operations with violation triggers, transfers and time windows. The integration of these subjects with time-dependent travel times and robustness of solutions makes this project of potentially very much added value.

### 2.6.2 Methodology conclusion

Based on the literature we described in this chapter we decide on a solution method that consists of three phases: triggers, search space selection and optimization of the solution. Triggers are developed to signal a need for rescheduling calculations. The triggers are a part of the rescheduling solution in general, they influence the moment of rescheduling, therefore influencing the results of a solution. However, not all vehicles on the road are relevant for a rescheduling solution. Variables such as distance to the disrupted vehicle and available capacity in the potential helper vehicle can be important in determining which potential helper vehicles can be part of the rescheduling solution.

We use a metaheuristic to optimize our rescheduling solution. Metaheuristic can be used to guide the local search within our search space to find a good solution more quickly, which is very important in our research. We select Tabu Search as the metaheuristic used for our research problem because of the quality, flexibility, and speed in finding feasible solutions (as described in Section 2.2).

Within the Tabu Search we have to select operators to generate neighbor solutions. From this chapter follows that the type I1 insertion heuristic by Solomon (1987), based on the minimization of extra distance and extra time required to visit the customer that is evaluated for insertion, is superior to other insertion operators. This can be used for the evaluation of transfer locations, which have to be inserted into routes to be able to transfer goods between vehicles.

For other operators that generate neighbor solutions to optimize routes, both intra- and inter-route, we selected the exchange, 2-opt* and Or-opt operators. We use the exchange operator because it is simple, effective and useable for both intra- and inter-route optimization. We select 2-opt* and Or-opt operators because they are well adapted to problems with time windows and because they preserve the orientation of routes.

# 3 PROCESS- & DATA-ANALYSIS

This chapter gives insight into the case study process and data. We aim to find the current process of rescheduling and we give an insight in the case study data. Section 3.1 presents information on the current processes in the case study. Section 3.2 gives an analysis of case study data. Section 3.3 describes a chapter conclusion on the process and data of our case study.

## 3.1 PROCESS ANALYSIS

As mentioned in the introduction, our case study is a Dutch retailer with a home delivery service. They use 14 different departure locations within the Netherlands, referred to as depots in the remainder of this study. All vehicles that deliver goods for the retailer belong to a given depot where they start and end their route.

All trips made by the vehicles are monitored and controlled by a control room. Now, potential delivery problems as described are observed by the vehicle drivers or the control room team. The problems and solutions are communicated by phone. As mentioned in the introduction, the control room team basically uses three different rescheduling strategies: 1) Switch, 2) Transfer, and 3) Accept & inform. These strategies mean that they 1) alter the sequence of customers within a route, 2) transfer customer orders to another vehicle, and 3) accept the expected lateness and communicate this towards the customer. Backup vehicles are sometimes used for transfers as well. It has been a management choice not to use too many backup vehicles because of the cost. Therefore, three vehicles, located in Rotterdam, Utrecht and Almere, are used as backup vehicles.

The control room team tries to limit the number of Time Window (TW) violations during the execution of a planning instance. We define a TW violation as a delivery that is started when the time window has already ended. Only the late deliveries are set out because a vehicle waits when it arrives too early. The planning instances used by the retailer are based on minimization of cost. In a cost minimization, the number of vehicles used is also an important factor. The retailer cannot use a separate vehicle for each customer because this would be too expensive. Therefore, they aim to use all available vehicles as much as possible in the delivery process, meaning little slack is planned for travel and service times. Unexpected delays can ensure TW violations later in the delivery process.

Figure 5 displays the case study workflow timeline. The following tasks are presented for both the PO morning (PO) and afternoon (PA) shifts on separate timelines in this figure:

1) Planning is made by the retailer;
2) Planning is uploaded into the DCT;
3) Vehicle numbers are linked to planned routes;
4) Vehicle numbers are provided to the DCT;
5) Vehicle is loaded at the depot and a sheet with the planned route of the vehicle is provided to the driver;
6) Start of execution of planned routes and start of the control room team monitoring the vehicles;
7) Rescheduling of disrupted routes.

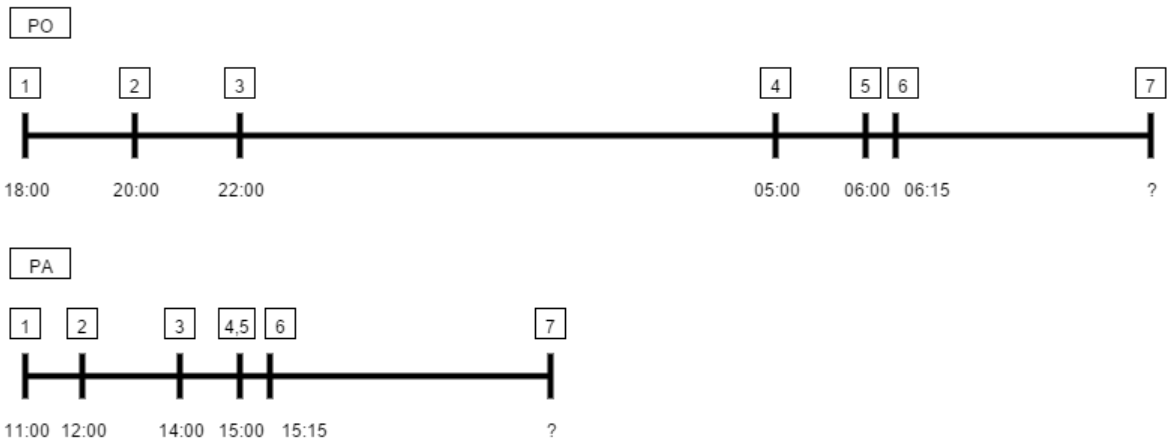


*Figure 5: workflow timelines of current planning and driving process.*

The control room team monitors the home delivery and they possibly notice delays from their monitoring in the DCT. It is also possible that vehicle drivers call the control room about a delay. The way in which the control room team determines a rescheduling of routes is presented in Figure 6. Here, the disrupted route is the route that is expected to be late for one of its deliveries.

Work agreements of the retailer state that drivers call the control room instantly when a breakdown of their vehicle occurs. Furthermore, drivers call the control room when they are 20 minutes or more behind on schedule. The control room team checks for rescheduling options within the disrupted route itself (intra-route) first. When a call arrives, the team checks if there is a problem for one of the customers. When this is not the case, they continue as currently planned. Otherwise the team checks if they can solve the problem by moving the stop forward in the sequence, meaning that the possible problem stops receives its delivery earlier in the sequence than originally planned. When the control room team cannot solve the problem in this way, they check for empty vehicles near the disrupted route. They only use empty vehicles to reduce the risk of disrupting the planned deliveries of other routes. When an empty vehicle is found near the disrupted vehicle, the control room sends the empty vehicle to meet the vehicle of the disrupted route to transfer some of the deliveries. Back-up vehicles and an available driver wait at three of the larger depots. When necessary, these backup vehicles can be used by the control room team to help out disrupted vehicles as well. If no empty vehicle is available near the disrupted vehicle that could ensure on time deliveries, all customers that are expected to receive a late delivery are informed by the customer service.

The control room team generally reschedules routes only once in the current working process. Furthermore, a distinction is made between new and recurring customers when checking priorities. A customer is considered to be new only the first time a customer orders at the retailer. New customers have a higher priority for on time delivery when rescheduling is performed. This means that the new customers are always considered first for helper actions when having to choose between customers that receive their goods outside their time window.

A lot of manual work is involved in this current rescheduling process, resulting in many possibilities for human errors. On top of this, only empty vehicles are now used to help out on the disrupted routes. The possibility of other vehicles that help out are topic of this research as well.

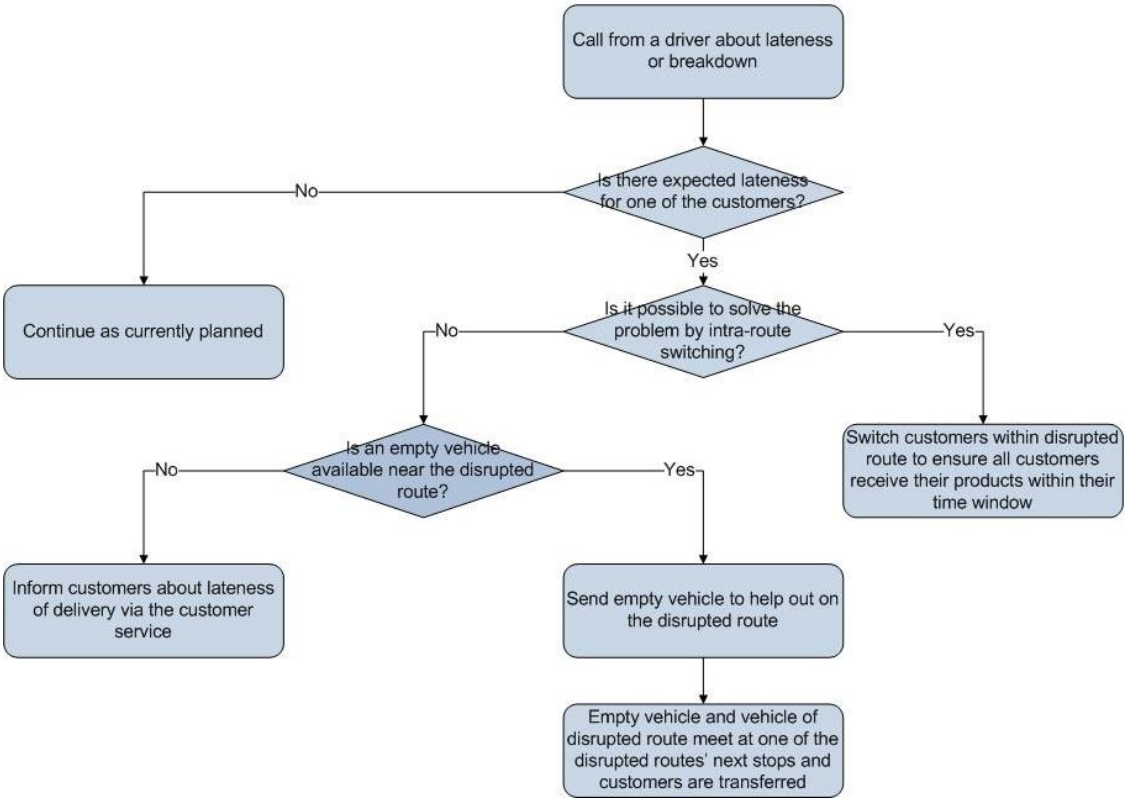


*Figure 6: Current rescheduling operations as performed by the control room of the case study retailer.*

## 3.2 DATA-ANALYSIS

With our data-analyses in this section, we aim to give an insight in different activities and durations in the delivery process for our case study retailer. Because of the recent introduction of the DCT and the data confidentiality of the retailer, a limited period of time is used. We use data for the time period of September 7$^{th}$ to October 11$^{th}$ in this section. An analysis of lateness of delivery is included in Appendix A. We also provide other analyses there that are not included in the main report because of data confidentiality. In this section, we first describe the departure and service times of our case study. Next, we analyze travel times and a travel time approximation for the case study.

### 3.2.1 Departure and service times

We introduce analyses of departure and service times registered in the DCT here. We end this part of our data-analysis section with a short conclusion on departure and service times.

#### *3.2.1.1 Departure times*

We analyze the lateness of departure from depot to see if the depot departure lateness is a cause for lateness of delivery. In the researched time period, we find the results shown in Figure 7. We show the lateness of departure class, where we subtract the planned departure time from the realized departure time, on the x-axis and we show the percentage of trips that are in this lateness class on the y-axis. We see that vehicles generally departure too late from the depot. The biggest percentage of vehicles departs between 0 and 10 minutes late from the depot. On time departure could already prevent a lot of lateness in the delivery process, meaning it could also lead to more on time deliveries.

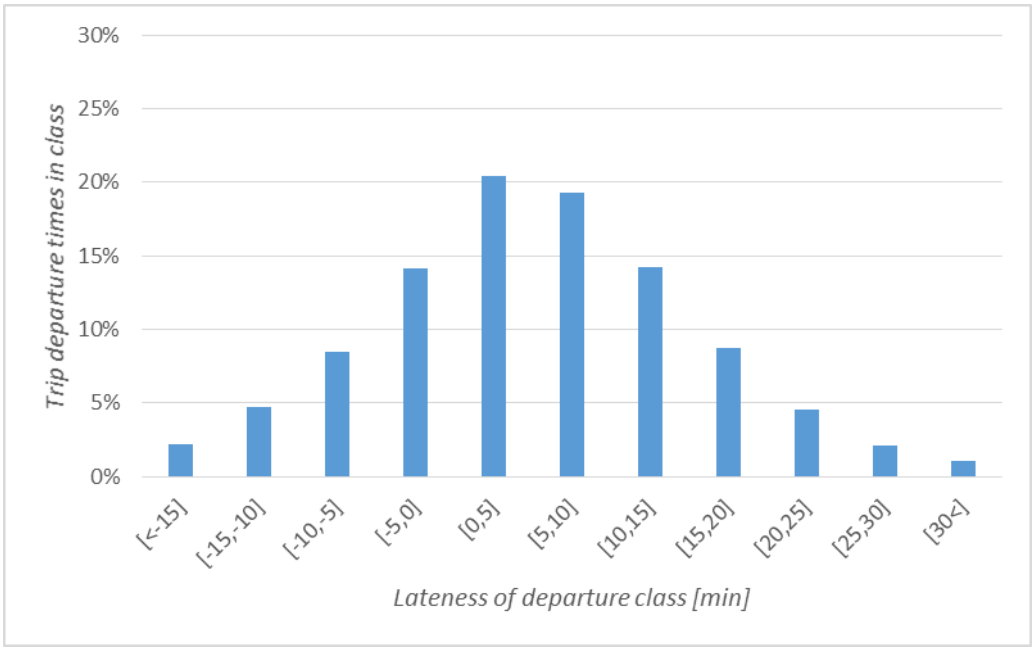


*Figure 7: lateness of departure from the depot for all registered trips in the DCT between 07-09-2015 and 11-10-2015.*

#### *3.2.1.2 Service times*

We analyze service time delays here to see the delays that occur during the service of customers. Figure 8 shows the results for the researched time period. The delay class is the number of minutes that a registered service time was longer than planned. We see that around 50% of the service times is realized in between 2 minutes too short and 2 minutes too long. The planned service times appear to give a good estimation for the service times, although small differences do occur. We see that delays in the service times follow a normal distribution and we conclude that the planned service time is generally a good estimation for a service time.

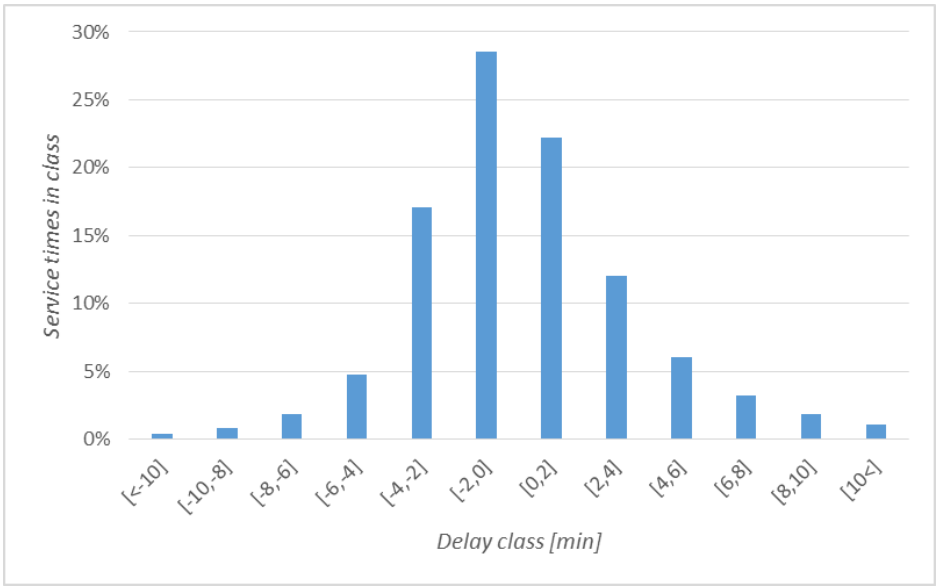


*Figure 8: service time delays for all registered service times in the DCT between 07-09-2015 and 11-10-2015.*

### 3.2.2 Travel times

We research travel times and differences with realized travel times here for planned travel times and Euclidian travel times. We analyze the first because we want to know the quality of planned times. We analyze the second because we want to have a fast alternative for travel time calculations in our simulation model (see Chapter 5).

#### *3.2.2.1 Planned travel times*

We analyze the planned travel times and compare them to the realizations from the DCT. Figure 9 shows that the planned travel times are not a good predictor of the realized travel times. We see much deviation in the realized travel times compared to the planned travel times. Furthermore, when vehicles drive the customers in a different order than originally planned, no access to the travel time algorithm used for the planning is available. We conclude that the planned travel times are not a good estimation for travel times, and that late deliveries can partly be explained by the travel time deviations that occur in the delivery process.

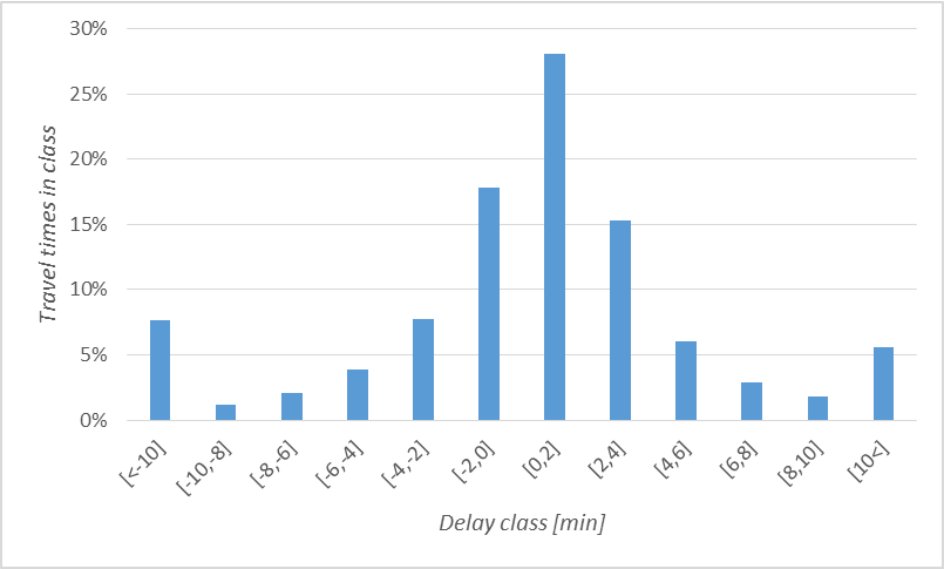


*Figure 9: travel time delays for all registered travel times where the vehicles drove in the planned order in the DCT between 07-09-2015 and 11-10-2015.*

#### *3.2.2.2 Euclidian travel times*

We analyze an approximation of travel times here for when fast calculation of travel times is necessary. In a simulation context, real travel time calculations mean larger computation times, resulting in large computation times for the simulation runs as well. We propose and analyse an approximation where we use a freeflow travel time – the travel time of a vehicle that could drive freely everywhere and would experience no delays (traffic and non-traffic) whatsoever – and a delay compared to this travel time. We research the subject in different distance classes because travel time deviations can differ a lot on different distances, especially when analyzing the delays with percentages.

For the freeflow we use the 95$^{th}$ percentile of the speeds for each of the distance classes, where speeds are calculated by taking the Euclidian distance between two stops and dividing this by the registered travel time. We listed the freeflow and mean travel times in the different distance classes in Table 1. Appendix B presents figures of the distributions we find for the realized percentage delays compared to the travel times based on the freeflow speeds for Euclidian distances. We find feasible values for both the average and freeflow speeds, while a logical pattern occurs where average speeds are much lower when the distance between stops is lower. We conclude that this Euclidian travel time approach is realistic enough to use.

*Table 1: realized travel time and speed values for different Euclidian distance classes.*

| Distance class [km] | Mean travel time [min] | SD travel time [min] | Average speed [km/h] | Freeflow speed [km/h] |
|---|---|---|---|---|
| 0.0-0.5 | 2.94 | 2.00 | 5.61 | 13.75 |
| 0.5-1.0 | 5.92 | 2.61 | 8.71 | 16.61 |
| 1.0-1.5 | 7.63 | 3.04 | 11.15 | 19.36 |
| 1.5-2.0 | 9.04 | 3.27 | 13.06 | 21.52 |
| 2.0-2.5 | 10.27 | 3.45 | 14.73 | 24.10 |
| 2.5-3.0 | 11.33 | 3.79 | 16.39 | 26.40 |
| 3.0-4.0 | 12.73 | 4.28 | 18.45 | 29.58 |
| 4.0-5.0 | 14.31 | 4.59 | 21.08 | 32.80 |
| 5.0-6.0 | 15.96 | 4.92 | 23.01 | 34.66 |
| 6.0-7.0 | 17.63 | 5.49 | 24.66 | 36.83 |
| 7.0-8.0 | 18.96 | 5.80 | 26.42 | 38.76 |
| 8.0-10.0 | 20.79 | 6.00 | 28.57 | 42.35 |
| 10.0-12.5 | 22.94 | 6.05 | 31.32 | 45.21 |
| 12.5-15.0 | 25.37 | 6.57 | 34.60 | 49.00 |
| 15.0-20.0 | 29.62 | 7.75 | 38.06 | 54.60 |
| 20.0-25.0 | 34.17 | 8.45 | 42.16 | 58.32 |
| 25.0 < | 41.03 | 11.13 | 50.39 | 70.38 |

## 3.3 PROCESS- & DATA-ANALYSIS CONCLUSION

We describe the current process of rescheduling and insights on the case study data in this chapter. We find that a labor-intensive and simple rescheduling procedure is used by the case study retailer. Drivers call the control room and they determine en-route rescheduling solutions to reduce the number of late deliveries. They use three basic strategies to do so: 1) intra-route switching in the sequence of customers, 2) inter-route helper actions, where a helper vehicle drives towards a disrupted vehicle to take over some of its customers, or 3) accepting the late delivery and inform the customers about the lateness of delivery. They use relatively simple ways to determine their solutions, only utilizing moving expected late deliveries forward in the sequence or using empty vehicles as helper vehicles.

From our data-analysis we find that late departure from the depot is an important source of delays on route. The effects of on time departure are interesting to research. The planned service times seem to give a good prediction of service times. No structural delays can be found in the service times. The planned travel times do not give a good approximation of travel times and planned travel times are not always available for all paths. Therefore, we propose an approximation based on Euclidian distances. The approximation appears to be realistic enough to be used in a simulation model

# 4 RESCHEDULING METHODOLOGY

This chapter describes the methodology we propose for en-route rescheduling of home deliveries. Aim is to set out the methodology for automated en-route rescheduling based on the findings from our literature review and process- and data-analysis. Our rescheduling algorithm consists of three steps that follow from Chapter 2: triggers, search space selection and a rescheduling algorithm. Section 4.1 introduces the triggers that are relevant for a home delivery operation. The triggers are a part of the rescheduling solution in general; they influence the moment of rescheduling, therefore influencing the results of a solution. Section 4.2 describes the process where we limit the search space to only the most relevant vehicles that might be able to help a disrupted vehicle. Section 4.3 describes the design of our metaheuristic for rescheduling. Section 0 ends this chapter with conclusions on the methodology.

## 4.1 TRIGGERS

This section introduces the triggers used in our solution methodology. Within the Delivery Control Tower (DCT) of Simacan, all current locations, planned deliveries and real-time calculated Estimated Times of Arrival (ETA's) are known, and these variables are updated every minute. The Simacan routing algorithm, TomTom real-time travel times, and TomTom profile travel times are used to determine the ETA's in the DCT, using the method as described in the introduction. At each new vehicle position we can determine whether or not a rescheduling operation is relevant as well. When this is the case, a rescheduling calculation will be performed and possible en-route changes can be suggested. We base the following list of triggers on all real-time information sources available in the DCT:

- **Delay trigger**: current delay of a vehicle on its route is larger than a threshold delay. This threshold delay can be determined using the smallest difference between planned service start time at a customer and the end of the time window. Other threshold values are possible as well however, for example the difference mentioned with an additional five minutes.
- **ETA trigger**: the ETA of the vehicle (calculated in the DCT) is later than the end of the time window as defined by one of the customers. This means a delivery is expected to be too late, which is undesirable for the retailer. Keep in mind that an ETA trigger can replace the delay trigger, since this trigger predicts expected times based on occurring delays as well. This trigger should only go of when no other path to prevent a delay is possible, since choosing another path from A to B is also possible for traffic-related delays.
- **Robustness trigger**: this type of trigger applies when the robustness of a route is in danger, meaning that the probability of possible lateness further on in the route is large. The robustness trigger can apply, for example, when a vehicle has two or more stops where it expects to arrive in a 5-minute window around the end of a customer time window. The robustness trigger is an addition to the delay or ETA trigger.
- **Vehicle defect trigger**: this trigger applies when a small defect of a vehicle ensures that the vehicle cannot continue right away. A roadside assistance vehicle is send to resolve the defect at the place where the vehicle is stranded. The fleet support manager of the retailer states that this type of defect occurred 773 times for the approximately 135,000 trips made in 2014. Average delay time is about one hour on average (45 minutes of waiting time for the roadside assistance vehicle, 15 minutes of repair time).
- **Breakdown trigger**: a breakdown of a vehicle occurs during the route of a vehicle. The vehicle is stranded and has to be towed. The deliveries that are in the trucks are either not delivered to the customer, or a helper action is performed by another vehicle that takes over all

customers of the stranded vehicle. The fleet support manager of the retailer states that this type of defect occurred 87 times for the approximately 135,000 trips made in 2014.

- **Accident trigger**: there is an accident on the road that interferes with the route of a vehicle. An accident is expected to affect an area where multiple vehicles might be present or planned to drive. When no rescheduling or the choosing of alternative paths is used, all vehicles are expected to experience delays.
- **Event trigger**: there is an event (e.g., soccer matches or carnival parades) that ensures congestion on the roads where retailer vehicles drive or are a planned to drive. This triggers a rescheduling operation when the event ensures large delays, and when it is not expected to be resolved quickly. Event delays are expected to affect an area. The event trigger is a different type of trigger from the other ones mentioned here. It is possible to know about large events before the start of a shift, meaning the retailer can plan differently instead of using en-route rescheduling. Still, events can be unnoticed till the actual day of delivery (e.g., larger block parties or small festivals), making it necessary to reschedule en-route.

As said, the triggers can be checked at each new vehicle position. A trigger is followed by a rescheduling calculation. It is possible that triggers for multiple vehicles are active at the same time because vehicle positions are generally sent to the DCT in batches. When multiple triggers go off at the same time, multiple rescheduling calculations can start at the same time. Of course, when these rescheduling calculations have a search space with no overlapping vehicles, the rescheduling calculations can be done parallel to each other. However, when the rescheduling calculations do have an overlap in vehicles, the solutions should not conflict with each other. Therefore, the solution would be to consider one rescheduling calculation with multiple disrupted routes. The number of occurrences of this case is expected to be very small because the triggers would occur on exactly the same minute. The case when multiple disrupted routes occur at the same time is therefore not considered as topic in this research.

Triggers can possibly keep on giving trigger signals for the same route because we determine new positions and delays every minute. However previous rescheduling calculations might already solve the problem or determine that no better option is available. Therefore, we recommend the use of a “rescheduled” state to keep track of which vehicles have just been rescheduled. When either no better option is available or when an intra-route switch is made, the route and vehicle remain in a “rescheduled” state for a given period of time (for example an hour). When a transfer between vehicles is made, both vehicles remain in a “rescheduled” state until all transfer actions are finished. In calibration of this part of our rescheduling methodology we tune the triggers in a way that we do not want the system to be too nervous, meaning that the system would trigger rescheduling calculations too often. We tune only the ETA and robustness triggers in this process.

## 4.2 SEARCH SPACE SELECTION

This section outlines the selection of a search space to reschedule routes. The selection of possible helper vehicles that can play an important role in finding good solutions more quickly. We propose a method to limit the search space in order to speed up the search process in a real-time context. We allow pre-emption when the current activity of the vehicle is driving, waiting or breaking. A vehicle is always obliged to finish its service activity at a customer, meaning that no pre-emption is allowed. The service time has to be finished by the driver because he cannot stop in the middle of a delivery.

As found in our literature review in Chapter 2, several factors as time of disruption, vehicle location and potential backup vehicles are important in selection of a helper candidate. Because there are multiple factors that determine whether or not a vehicle is available to be rescheduled, we define an equation to select the vehicles other than the vehicle of the disrupted route (which is always in the

search space) in a search space for home deliveries. We use the following variables in the search space selection equation:

1) Current location of the potential helper vehicle is the first of our selection variables. Distance from the potential helper to the disrupted vehicle is an important attribute in this context. This is important because the distance relates to the additional travel time needed for to be able to transfer customers. We use a Euclidian distance (in meters) for this distance because this enables us to determine many distances fast.
2) When a vehicle has to make deliveries for many customers itself, it is not smart to use this vehicle in a helper action. There is still too much unknown about its own route, ensuring that the transfer might be a bad idea because we only shift a problem to a different route. We therefore use the number of customers that a potential search space vehicle still has to visit itself as an additional variable for in- or exclusion in the search space.

We want to ensure that vehicles that are either too far from the disrupted vehicle or that have too many customers in their own route are excluded from the search space. We therefore set two threshold values for both variables. We exclude vehicles that have one of the search space selection variables that exceeds the threshold value. We set the following threshold values for our search space selection: 1) the distance between of helper vehicle to disrupted vehicle is 10 kilometers or less and 2) the number of customers to deliver for the helper vehicle should be 4 or less.

We use the definitions from Table 2 to determine a general function for a score of a vehicle in the search space selection process. The equation for the search space score of each vehicle is given in Equation 1.

*Table 2: variables used in search space selection equations.*

| Term | Definition |
|---|---|
| $K$ | Set of all vehicles. |
| $dv$ | Vehicle of the disrupted route. |
| $d$ | Distance between two vehicle locations. |
| $loc_k$ | Location of vehicle $k$. |
| $n_k$ | Remaining customers for vehicle $k$. |
| $\alpha_1$, $\alpha_2$ | Variables determining the importance of the factors in the search space selection. |
| $s_k$ | Score for vehicle $k$. |
| $S_{threshold}$ | Threshold score for inclusion of vehicle in the search space. The score of a vehicle has to be at least equal to this threshold score. |

$$s_k = \frac{1}{\alpha_1 * d(loc_{dv}, loc_k) + \alpha_2 * n_k} \qquad (k \in K - \{dv\}\,; \alpha_1 + \alpha_2 = 1) \qquad (1)$$

The score is used to determine which vehicles are in the search space of our rescheduling process. We first determine a score for each of the vehicles, and then select the vehicles relevant for rescheduling using $S_{threshold}$. Based on the threshold values for our search space variables as mentioned, we can determine the $\alpha$-values by setting the threshold values equal to each other. This means that we the equation defined in Equation 2. We find that the value for $\alpha_1$ is 0.28 and the value for $\alpha_2$ is 0.72 (where the sum is equal to 1). Using these values, we come to a $S_{threshold}$ of 0.35 for inclusion in the search space.

$$\frac{1}{\alpha_1 * max\_dist + 0} = \frac{1}{0 + \alpha_2 * max\_cust} \qquad (2)$$

## 4.3 RESCHEDULING ALGORITHM

We describe the rescheduling calculations that we develop in this study. As stated before, we use a metaheuristic to guide the search for a good and feasible solution for our research problem of home delivery routes. We use a metaheuristic because it is not possible to find the exact solution for the problem. Metaheuristics are suitable to find good solutions for large NP-hard problems, such as the VRPTW or subclasses of the VRPTW. In Chapter 2 we selected Tabu Search as the metaheuristic used for our research problem because of the quality, flexibility, and speed in finding feasible solutions. First, we describe the Tabu Search Rescheduling (TSR) algorithm we developed. After this, we introduce the objective function, possible operators and stopping criteria for our algorithm.

### 4.3.1 Tabu Search Rescheduling algorithm

We develop a Tabu Search implementation that evaluates options to solve our rescheduling problem by intra-route switching and by inter-route transfers simultaneously. The state of the routes involved in the rescheduling are defined as the initial solution in our rescheduling process. We choose an approach with intra-route and inter-route calculations because we want to solve occurring problems by continuing as planned or switching intra-route as much as possible. However, when no other options can solve the problem, we want to transfer orders between routes (inter-route) to ensure that as many customers as possible receive their orders on time. Our objective function determines the quality of each of the solutions, and selects the best. Here, a measure of the time needed over all vehicles is included as well to discourage vehicles from detours to help other vehicles.

Figure 10 displays our Tabu Search Rescheduling (TSR) method within the selected search space that follows from the method in Section 4.2. We introduce the intra-route part (left side of the figure) and inter-route part (right side of the figure) subsequently here.

#### *4.3.1.1 Intra-route Tabu Search Rescheduling*

As said before, we aim to solve as many of the rescheduling problems by intra-route changes because this has less impact on the whole delivery process than inter-route changes. We start with the current state of the disrupted route as the initial solution. This is also initialized as the current solution to start our search process and we initialize an empty Tabu list. In the Tabu list we will keep track of the solutions we already evaluated to keep the local search from cycling at local minima. In the entire process we do remember the very best solution of the search to ensure that we eventually select the best solution visited.

After initializing an empty Tabu list we randomly select a move-operator from the allowed moves list to check the neighbor solutions. The allowed operators differ per experiment (see Chapter 5), but for most experiments multiple operators are allowed. When we have an operator, we evaluate all solutions that are direct neighbors of the current solution by checking the move for each customer in the search space where the resulting solution is not on the Tabu list. We evaluate all neighbor solutions using the objective function, selecting the best move as the move to carry out. We carry out this move on the current solution, thereby updating the current solution to a new one. We add the previous solution to the Tabu list to prevent the search process from visiting it again.

When we not yet satisfy one of our stopping criteria, we continue with the selection of a new operator and checking the neighborhood. We repeat this procedure until we reach the stopping criteria we defined for the intra-route TSR part. We set the best solution found during the entire search process as the best switching solution for the disrupted route. We evaluate this in context with the other routes in the search space to be able to compare the solution with the initial solution and the best transfer solution.

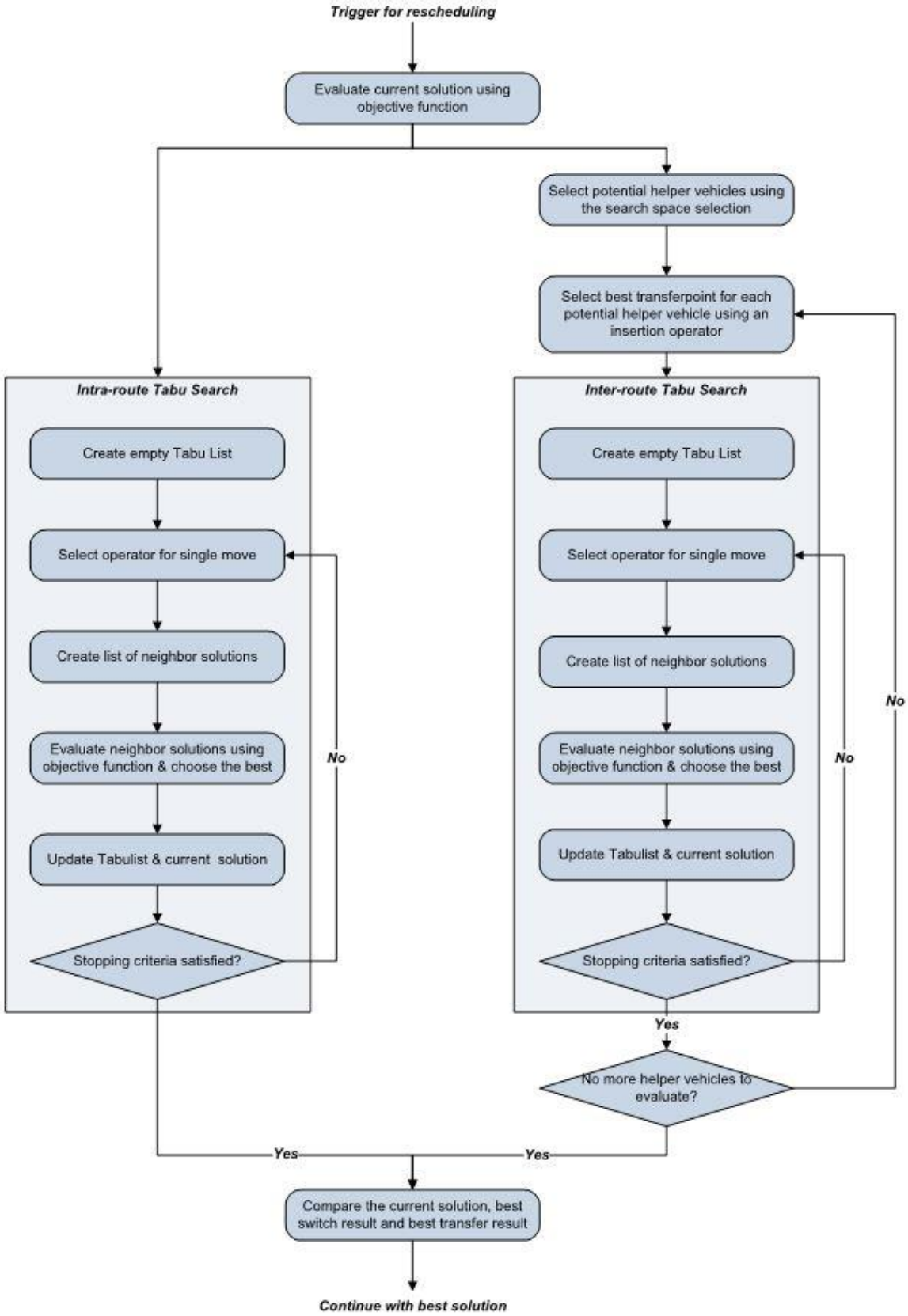


*Figure 10: Tabu Search Rescheduling (TSR) method as developed in this study.*

#### *4.3.1.2 Inter-route Tabu Search Rescheduling*

Simultaneously with the intra-route TSR part, we start with the inter-route part of our TSR method. Here, we also start with the current state of the routes as initial solution, but we evaluate inter-route changes involving the disrupted route. Therefore, we evaluate a search space objective function result instead of a route objective function result, as done for the intra-route part of the TSR method.

We start with the search space selection step as described in the previous section. It is important to note that transfer can include both backup vehicles (waiting at a depot) and en-route vehicles. Both types of vehicles are initialized in the search space selection when a trigger for rescheduling is triggered if their search space score is high enough to be included in the local search.

For each of the initialized helper vehicles in the search space we will evaluate what the best location is to transfer orders from one vehicle to another. Two transfer types can be distinguished for making a transfer in general. A transfer is either performed at an intermediate point for two trips, or at a stop of one of the trips. Figure 11 shows an example of the transfer types for two example trip-ids: 901 and 902. For the first transfer type, the additional travel time is for both vehicles from a stop (or the current vehicle location) to the transfer point and from the transfer point to the next stop, where we subtract the travel time between the current location and the next stop. For the second transfer type, additional travel time only applies for the helper vehicle, where the additional travel time consists of the travel time from a current location to the transfer point and from the transfer point to the next stop, subtracting the travel time between the stop, or the current location, and the next stop.

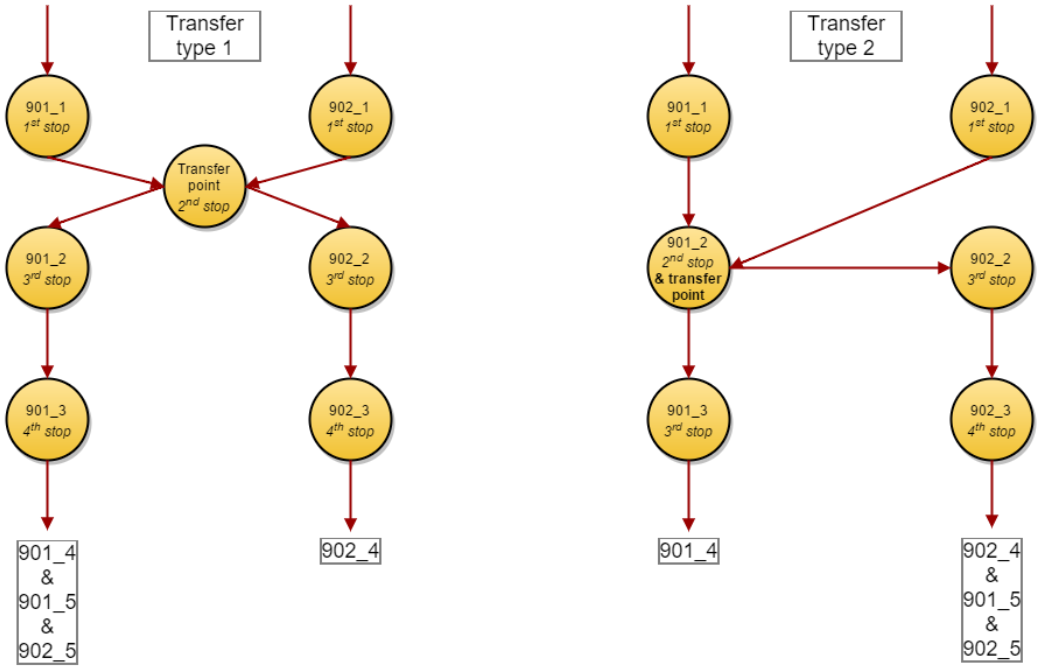


*Figure 11: Transfer types for en-route rescheduling.*

Because we do not want to evaluate all possible locations in the Netherlands for the transfers, we evaluate transferpoints in three classes for each of the potential helper vehicles. We select the best transferpoint found in this search based on an insertion heuristic we designed specifically for transfers. We first introduce the transferpoints we evaluate here:

1. *Meeting as fast as possible*: the two vehicles drive directly towards each other and they meet exactly in the geographical middle. When one of the vehicles is involved in a service time, meaning no pre-emption is allowed, the other vehicle waits a little longer at the transfer point.
2. *Meeting where paths cross*: when modelling the two routes as a graph, possible intersections between the helper-route and disrupted route exist. We calculate the transfer location using a general line intersection equation and evaluate this transfer between the stops where the line intersection occurs.
3. *Meeting at one of the stops of the disrupted route*: we evaluate all stops of the disrupted route as shown for the second transfer type (Figure 11). We only evaluate the stops of the disrupted routes because we do not want to further disturb the deliveries of this route.

The insertion heuristic we use to evaluate which of the transferpoints is the bested is based on the Solomon I1 insertion heuristic, because we found in Chapter 2 that this is a simple and efficient way to evaluate insertions. However, we modify this heuristic to make it applicable for our situation.

We define $u$ as the transferpoint we evaluate for insertion on a route. We define $d_{ij}$ as a distance between stop $i$ and $j$, $b_j$ as the time of service for a customer $j$, $b_{j(u)}$ as the new time of service for customer $j$, and $b_u$ as the time of service for the inserted transferpoint. Equation 3 gives the cost component for the additional distance travelled. Equation 4 gives the cost component for changed time of service for customers.

Because we insert the transferpoints in two routes we also calculate a waiting time. When one of the vehicles arrives before the other. We add the time it has to wait to the cost of the insertion in Equation 5. We multiply this cost with the same weight as the other time component. We evaluate the added cost for the helper-route ($k_h$) and disrupted route ($k_d$) and select the minimum insertion cost as the transferpoint for the helper-route we evaluate. Equation 6 combines the cost components from Equation 3, 4 and 5 to calculate insertion cost $c$, where $\alpha 1$ and $\alpha 2$ are used to determine weights for distance and time respectively.

$$c1_k = d_{iu} + d_{uj} - d_{ij} \qquad \text{For } k_d \text{ and } k_h \qquad (3)$$

$$c2_k = b_{j(u)} - b_j \qquad \text{For } k_d \text{ and } k_h \qquad (4)$$

$$c3 = \left| b_{uk_d} - b_{uk_h} \right| \qquad (5)$$

$$c = \alpha 1 * \left( c1_{k_h} + c1_{k_d} \right) + \alpha 2 * \left( c2_{k_h} + c2_{k_d} + c3 \right) \qquad \alpha 1 + \alpha 2 = 1; \alpha 1 \geq 0; \alpha 2 \geq 0 \qquad (6)$$

A lot of parameters can be altered in our TSR method and we have only limited time to do so. Therefore, we fix the weights of distance and time in this insertion heuristic here. We measure our distances in meters and our times in seconds for this evaluation. We mostly base all of our evaluations on time, and because the time windows are the whole reason we reschedule, we also want to focus on the time part of this insertion heuristic here. Therefore, we set the weight of the time at four times the weight of the distance. Because the total of the two weight has to be equal to 1, this means we set α1 to 0.2 and α2 to 0.8.

With the transferpoint as a known stop for both the potential helper vehicle and the disrupted vehicle we continue with the next step of our inter-route part of the TSR method. To determine the customers we transfer we use inter-route Tabu Search with all customers that are after the transferpoint for the helper-route and the disrupted route. As done for the intra-route Tabu Search, we first initialize an empty Tabu list and we select a move-operator for the neighborhood evaluation. With the selected move from the allowed inter-route moves we select the best solution using our objective function and we update our Tabu list. We repeat this process until the stopping criteria for the inter-route TSR are

satisfied. When we satisfy these stopping criteria we check if there is another vehicle to evaluate. When this is the case, we start selecting the best transfer location for this vehicle, otherwise we select the best transfer to be made over all potential helper vehicles and select this as the best transfer solution.

### 4.3.2 Objective function

Within the Tabu Search metaheuristic, we use an objective function to determine the best solution for the rescheduling problem. We define time in minutes for all routes as our main criterion for minimization. We use expected travel, service, break and wait times here because we deal with a future situation.

We minimize the time needed to visit all customers in our search space. Since we deal with a problem where time windows are important, we penalize time window violations in some way in our objective function. Furthermore, we give a bonus to solutions that are more robust against future uncertainties. The robustness of a solution is measured in the number of minutes between the end of a customer time window and the planned time of service. Both bonuses and penalties are expressed in minutes, so the result of the goal function is in minutes as well. Keep in mind that this score does not directly translate to the duration of the tour.

All routes have to return to their original depot. Therefore, we include the depot as the final stop of all routes. The origin locations of the routes are always given by the current location of the vehicle, where an attribute indicates the earliest departure time from the current location. All customers not yet visited are included as stops in the route. Within the minimization, all customers in the search space have to receive their order from one of the vehicles and vehicle capacities cannot be violated when transfers between vehicles are made.

In our objective function, we define $V$ as the set of all stops to be visited by all vehicles in our search space; customers, transferpoints and finally the depot. The planned arrival time at stop $i$ is defined as $x_i$. Furthermore, we define $t_i$ as the travel time towards stop $i$, $b_i$ as the break time of the vehicle when a planned break occurs in the drive to stop $i$, and $s_i$ as the service time for the activity at stop $i$ (delivery at a customer, transfer time at a transferpoint). Time windows are indicated by $a_i$ as the start of the time window, and $e_i$ as the end of a time window. Furthermore, we introduce waiting time $w_i$ when a vehicle would arrive at stop $i$ before the start of the time window. It then waits till the time window starts, because it is allowed to start at that time. A transferpoint is visited by two vehicles that have to be at the transferpoint at the same time. Therefore, waiting time is introduced for the first vehicle that waits for the arrival of the second vehicle. The objective function for our metaheuristic is defined in Equation 7.

$$\sum_{i \in V} t_i + b_i + w_i + s_i + f\_penalty(x_i - e_i) - f\_robustness(x_i - e_i) \tag{7}$$

Functions are used in the objective function to penalize time window violations (*f_penalty*) and to give a bonus to robust solutions (*f_robustness*). Both have as input the end of the time window subtracted from the planned arrival time, indicating the expected lateness of arrival. We experiment with both the time window violation penalty function and the robustness function in this study.

For our time window violation penalty we define three different types of functions that we want to research: 1) a linear penalty function, 2) a quadratic penalty function, 3) hard time windows with some 'grace' time period, where we allow arrival during a small time period after the end of the time window. In the first type of function, we penalize al time window violation linearly for every minute of violation.

Disadvantage of this type of function might be that it can compensate many possibly small violations by creating one large time window violation. We use 10 times the TW violation time as our function to ensure that TW violations are not easily compensated by reduced total route time. The second function type (quadratic) test a function that deals with this problem. Disadvantage here might be that many customers receive deliveries a little late. To keep the quadratic function in the same order of magnitude as the other penalty functions, we use a simple quadratic function of $x^2$. Equation 8 and 9 define the first two penalty function types.

$$f\_penalty(x) = 10x \qquad [0, inf] \qquad (8)$$

$$f\_penalty(x) = x^2 \qquad [0, inf] \qquad (9)$$

We experiment with two more penalty functions: 5 minutes 'grace' and 15 minutes 'grace' to see the effects of some additional time in a real-time rescheduling context. Hard time windows are forced using a large penalty value. A hard time window is modelled with a penalty value of 1,000,000 minutes instead of infinite minutes. This way, when no solution is found without violation a hard time window, we are still able to find the best solution compared to the others. However, if one solution ensure no violation of hard time windows, this solution is always best because of the large penalty value. 5 and 15 minutes are used as 'grace' values to see the effects of small (5 minutes) and somewhat larger (15 minutes) 'grace' time.

Because we work with a distinction between new and recurring customers in our study, where new customer time windows are indicated to be more important, we define hard time windows without 'grace' for all function types. However, we also experiment with no distinction between the customer types (as we will explain in Chapter 5). Figure 12 shows all types of time window violation penalty functions we test in this study visually.

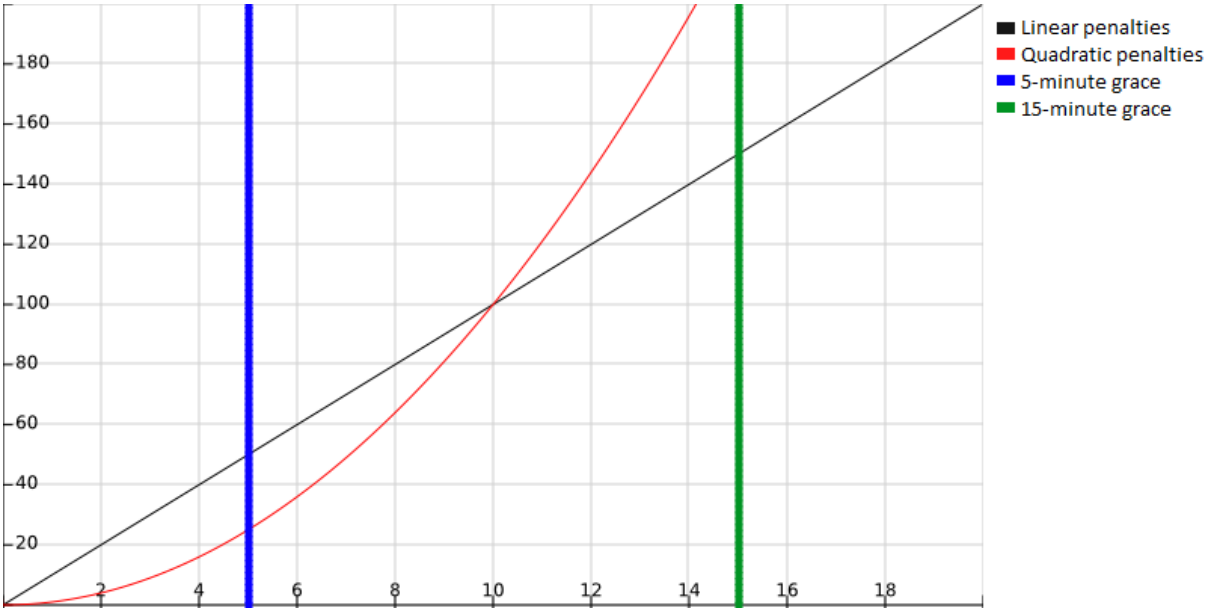


*Figure 12: time window violation penalty functions used in objective function.*

In our study, we also experiment with a robustness bonus function to find solutions that are more robust against future uncertainty. We use a function that gives a bonus only when negative delays exist. The maximum value of the bonus function is defined as *max_robustness*, a value we will fix in calibration of our rescheduling algorithm. Because a solution is not much more robust from other

solutions at a certain input value, we give the maximum robustness bonus when there is at least 15 minutes between the end of the time window and the planned arrival. We do not use a linear function between 0 and 15 minutes of robustness because a robustness increase from 1 to 2 minutes is more valuable than a robustness increase of 14 to 15 minutes. Therefore, an exponential type of function is used to calculate bonus values on the interval between 0 and 15 minutes of robustness. The exponential function is calibrated visually using an online function plotting tool. Rapidly increasing bonus for the first part of the bonus function, while flattening the bonus for the larger robustness values to the *max_robustness* value, are calibration goals for this function. We find the bonus function defined in Equation 10 to fit this purpose. Figure 13 shows the robustness function tested in this study visually.

$$f\_robustness(x) = max\_robustness * (1 - 2^{-0.5*x}) \qquad [-inf, 0] \qquad (10)$$

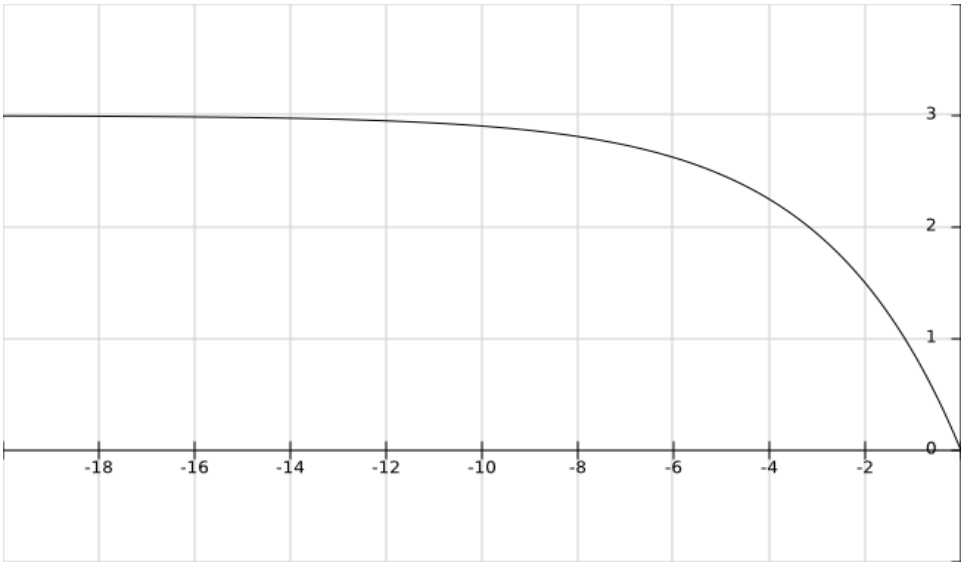


*Figure 13: Robustness bonus function used in objective function.*

### 4.3.3 Rescheduling operators

Several operators can be selected for finding the best routes for intra- and inter-route Tabu Search. We describe the operators we selected in Chapter 2: exchange, 2-opt*, and Or-opt. The 2-opt* and Or-opt operators where found in the literature as the best heuristics for problems with time windows. The exchange operator is included as a very fast and simple alternative. In the next chapter we experiment with the use of one or more of the operators.

#### *4.3.3.1 Exchange operator*

The exchange operator simply switches two customers that are within the 'local' Tabu Search process. It can be used for both intra-route and inter-route exchanges (both vehicles of inter-route exchanges have to be involved in the transfer). Major advantage of this heuristic is that the implementation is simple and fast. Major disadvantage is that is does not consider time windows in selection of the exchange candidates, so that it often evaluates options that are not likely an improvement on the current solution.

An example of an intra-route exchange is presented in Figure 14. Here, stop 901_3 and stop 901_5 are exchanged in the sequence of delivery shown in 14a. The outcome of the exchange operator is shown in Figure 14b.

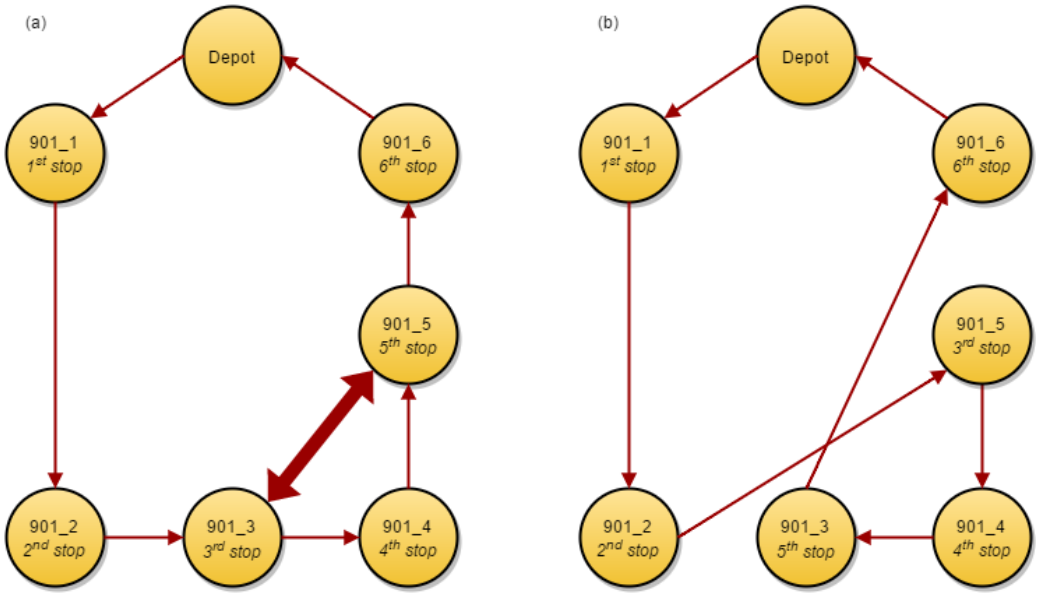


*Figure 14: example of an intra-route exchange operation where: (a) is the original situation, and (b) is the exchanged-nodes situation. Note that it is also possible to use these exchange operations for inter-route exchanges.*

#### *4.3.3.2 2-opt* operator*

2-opt* moves are specially designed for the use in problem instances with time windows. Problems with multiple routes enable us to exchange pairs of links while keeping the orientation within the route. In the 2-opt* heuristic, two arcs in the two selected routes are eliminated, replacing them with new arcs that connect the first customers on the first route to the last customers on the second route. An example is provided in Figure 15, where arcs 901_2 → 901_3 and 902_1 → 902_2 are eliminated and replaced by new arcs that connect the 901 and 902 routes.

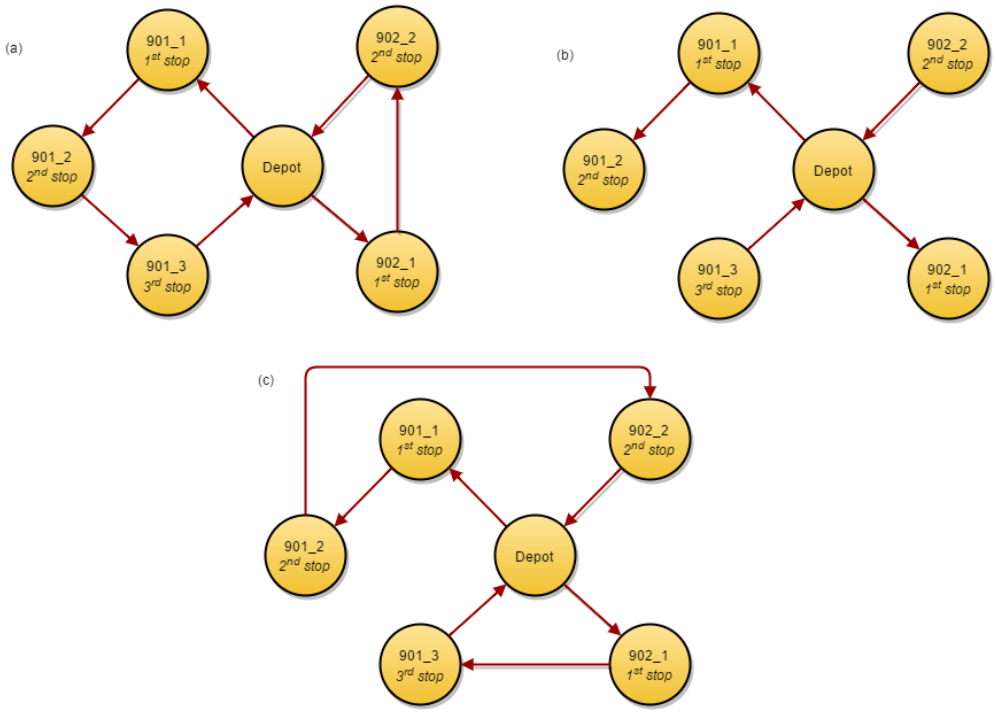


*Figure 15: example of 2-opt* neighbor-generating move where: (a) is the original situation, (b) is the eliminated arcs situation, and (c) is the replacement arcs situation.*

#### *4.3.3.3 Or-opt operator*

The Or-opt exchange is a well-known node exchange heuristic. Where k-opt heuristics consider breaking arcs and replacing them with new connections, the Or-opt exchange considers sequences of one, two and three adjacent customers in a solution, moving it back or forth in the route sequence. The Or-opt heuristic moves sequences of adjacent customers that are close in space and time. The customers are inserted at a new location, while the orientation of the routes is preserved. Or-opt exchanges are suitable for inter-route and intra-route modifications, and because we use a small number of routes in the Or-opt Tabu Search, we use it both intra-route and inter-route (both vehicles of inter-route moves have to be involved in the transfer). An example of the Or-opt heuristic is provided in Figure 16, where the sequence of stop 901_1 and 901_2 is moved in front of stop 901_4.

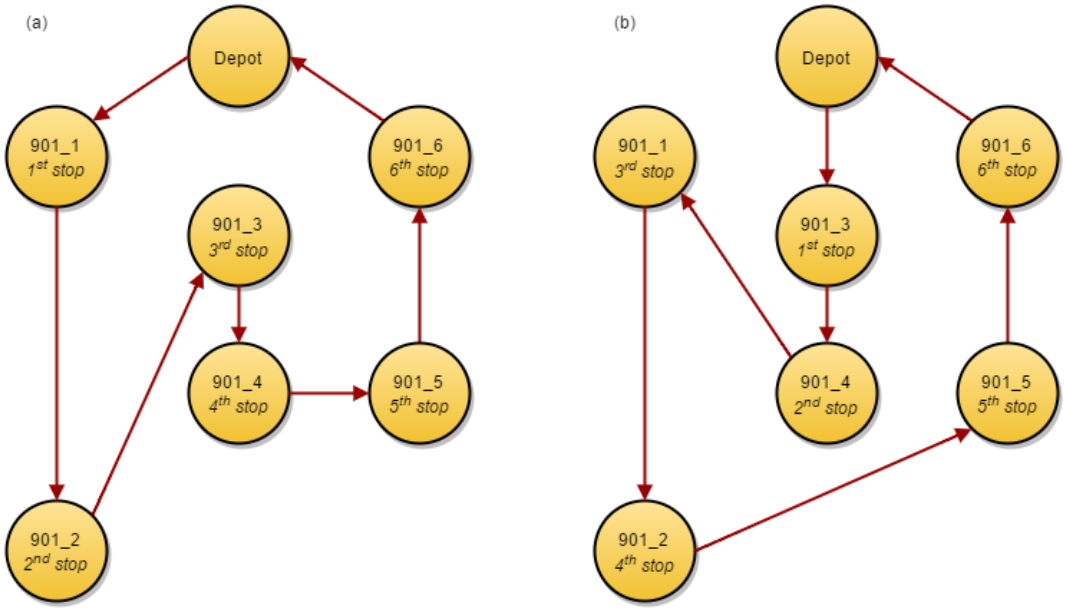


*Figure 16: example of an intra-route Or-opt neighbor-generating move where stop 901_1 and 901_2 are moved to behind stop 901_4. (a) Shows the before Or-opt situation, (b) shows the after Or-opt situation.*

### 4.3.4 Stopping criteria

The stopping criteria for our Tabu Search are defined as follows: if the best solution so far has not improved for $I$ ($I_{intra}$ and $I_{inter}$) consecutive iterations or when the total runtime of the algorithm is more than $T$ ($T_{intra}$ and $T_{inter}$) seconds, the we terminate the Tabu Search process and return the best solution so far. The stopping criteria are part of the experimental factors of our simulation setup, meaning that we experiment with the best settings for the stopping criteria.

## 4.4 RESCHEDULING METHODOLOGY CONCLUSION

Our TSR method to solve rescheduling problems consists of three phases: triggers, selection of a relevant helper vehicles, and rescheduling calculations. We identify possible triggers on delays, ETA's, robustness, vehicle defects, vehicle breakdowns, accidents and events. When we trigger for rescheduling, we determine relevant vehicles we want to evaluate for helper-actions based on helper distance, and the number of deliveries to be made by the helper. We select vehicles we evaluate for helper actions based on a threshold score. We evaluate all helper vehicles by insertion of the transfer location using an adapted version of the Solomon I1 insertion heuristic where we include waiting times. We test 2-opt*-, or-opt- and exchange-operators for inter-route Tabu Search to find the best inter-route solution given a transfer location we select. Simultaneously with the evaluation of inter-route transfers, we use Tabu Search to find the best intra-route switching solution.

We determine the solution quality of the initial solution, all possible helper actions and the intra-route switching using an objective function, and we select the best solution found to continue the routes of the en-route vehicles. The objective function consists of total time needed for the vehicles, time window violation penalties, and possibly an exponential robustness bonus function. To test the effectiveness of our TSR method, we need to set up a simulation that enables us to compare the current rescheduling process (as decribed in Chapter 3) with our TSR method (as described in this chapter). We describe the simulation model we use in the next chapter.

# 5 SIMULATION MODEL

We set up a simulation model to be able to compare the current rescheduling process and the TSR method. We use simulation because we are not able to test the rescheduling of routes in real-life because of cost, time and control of operations. We set up a simulation model to achieve the following goals: to calibrate general settings for the TSR method, determine the effectiveness of different trigger settings, find good objective function settings and determine the most effective neighbor operators.

The type of simulation we use is discrete event simulation. Discrete event simulation only models the points in time at which the state of the system changes (Robinson, 2004). More specifically, we apply the three phase approach. This approach classifies Booked (B) events and Conditional (C) events. First phase of the approach is to go to the next chronological event. Second phase is to execute all B-events. Third phase is to execute all C-events.

We use a discrete event simulation software package called Simpy. We chose Simpy – a Python library specifically designed for discrete event simulation – as our simulation software because it enables us to very quickly simulate a large number of discrete events. Another advantage is that, because we programmed our algorithm in Python as well, we limit the number of programs used for our research. Disadvantage of using Simpy is the fact that it has no animation component. Print statements of events enable us to debug and verify the model, and to follow the occurring events in a simulation run.

We aim to model reality as closely as possible in our experiments. We use actual planning data of our retailer case study as the planned instances for our simulation runs. We do this to be as close to real instances as possible, but also to ensure that we do not have to make an initial planning for each of the simulated days. We use planning data of September 28$^{th}$ to October 11$^{th}$ for our simulation runs. We run simulations for all planned shifts in this time period: Monday to Saturday, both morning shift (PO) and afternoon shift (PA). We model the delivery process as a graph of nodes (i.e., stops) and edges (i.e., paths between the stops), where travel times determine the time a vehicle needs to drive over an edge.

Section 5.1 introduces the conceptual model for our simulation setup. Section 5.2 describes the implementation of methods in the model. Section 5.3 describes the model verification and validation. Section 5.4 gives conclusions that follow from our simulation model chapter.

## 5.1 CONCEPTUAL MODEL

A conceptual model is a non-software specific description of the simulation model that is to be developed (Robinson, 2004). To describe our conceptual model, we introduce the model assumptions, objects, events, input and output.

### 5.1.1 Model assumptions

To be able to model our real-world delivery process, we have to make assumptions to simplify reality. We make the following assumptions for our simulation model:

General assumptions

- The delivery process is modelled as a graph of nodes (i.e., stops) and edges (i.e., paths).
- Customer orders are all placed and definitive. No changes to either the demand or the time windows are made during the simulation runs.
- Service cancellations are not allowed, all customers need to be served.
- A homogeneous fleet of vehicles is used.

- Backhaul is not included. All transportation materials (crates, boxes, etc.) that are returned by the vehicles can be folded and stacked to ensure that negligible space is taken by the products.
- No additional goods are available at the depots. This means that the goods available in a specific vehicle have to be delivered by the current vehicle, or be transferred to another vehicle that is available for this.
- Customers are assumed to always be at home at the time of delivery. Customers are also assumed to be at home when a vehicle is up to 15 minutes earlier than the start of the time window.
- Once a vehicle has returned to its depot, it is otherwise occupied and it is not available for helping out other vehicles.
- One backup vehicle is available at three depot locations in the Netherlands: Almere, Rotterdam and Utrecht. All backup vehicles are empty and they are able to help instantly if necessary.

Departure times, service times & driving times

- Depot departure delays are based on the realized departure delays. As explained in Chapter 3, we have a good dataset of departure delays for all vehicles driving. We use these in our simulation runs as well, without additional uncertainty. When no depot departure is registered for the trip, the planned departure is used without any delay.
- Service times are based on realization data as well, meaning no additional uncertainty is used in the simulation runs. When no service time is registered for the trip, the planned service time we use the planned service time without delay. Service times are always completed, meaning that the process cannot be interrupted during the servicing of the customer.
- Both departure time delays and service time delays are modelled as 'sudden' delays, meaning that the delay is known when the vehicle departs from the depot or stop location.
- Travel times between stops are calculated using the freeflow speed (see Chapter 3) for the distance class on the Euclidian distance between the two stops. Next to using this approximation of the freeflow speed we sample a percentage travel time delay from the empirical distributions in Appendix B. We do this using empirical data sampling described by Robinson (2004). We sample a delay class from the distributions and use a uniform distribution within each delay class to find the exact percentage delay.
- We assume not-correlated travel delays (between one drive and the next) because delays in the delivery process that is modelled can be caused by multiple factors such as traffic, difficulty of finding a parking space or administrative actions of the driver. It is therefore hard to determine when we can, and when we cannot assume correlated delays.
- Travel time delays are gradually added to a path travel time. E.g., when a delay of 150% occurs, all parts of the path take 150% more time than planned.
- When calculating rescheduling options, we assume that we can accurately predict the travel times towards the following five stops, meaning we deal with a prediction horizon of approximately 80 minutes. We sample percentage delays as described, and we apply the same percentage delay on all travel times towards possible next stops from the current location in our rescheduling calculations. This means that for each changed sequence of stops we apply the same delays to ensure we do not make an unfair comparison. For the remainder of the travel times on each of the routes we plan on the average speed in each distance class because we assume that we cannot fully predict these travel times.

Breaks & waiting time

- Vehicle drivers are obliged to take planned breaks because of driving time laws and regulations. We therefore assume that breaks are always held. A break is usually planned before a stop. When a rescheduling operation is performed, breaks are held at the stop closest to the originally planned time of a break. The breaks are linked to a vehicle, so when a stop is transferred, the break is not transferred.
- Breaks are held at a stop location, right before the service time of the stop.
- Break times are equal to the planned break times of the retailer.
- Waiting time is applied for a vehicle when it arrives more than 15 minutes earlier then the start of the time window. A vehicle is allowed to start a delivery when it is 15 minutes or less too early compared to the start of the time window.
- When needed for a rescheduling operation, wait and break events can be interrupted. The remainder of the waiting or breaking time is cancelled.

Breakdowns & defects

- We know instantly about vehicle defects and breakdowns upon occurrence.
- A vehicle cannot have a defect and a breakdown on the same route.
- Vehicle breakdowns are technical vehicle problems that cause a vehicle not being able to continue its route. We determine the vehicle breakdown probability in our simulation setup to be 87/135,000=0.06%, based on the data provide by the fleet support manager (Section 4.1).
- Vehicle breakdowns cause a pickup of the remaining customer orders needed at the breakdown location.
- We initialize before the start of a simulation if a vehicle will have a breakdown during its route. When this is the case, we determine the time of the breakdown for this vehicle. The moment of a vehicle breakdown is determined using a uniform distribution over all travel times of a route, since the probability of vehicle breakdowns is the same for every minute that the vehicle drives.
- Vehicle defects are problems with a vehicle that can be solved with some minor roadside assistance. The probability of a vehicle defect occurrence is 773/135,000=0.6%, based on the data provide by the fleet support manager (Section 4.1).
- Vehicle defects take 60 minutes to solve.
- We initialize before the start of a simulation if a vehicle will have a defect during its route. When this is the case, we determine the time of the defect for this vehicle. The moment of the vehicle defect is determined using a uniform distribution over all travel times of a route. The underlying assumption here is that the probability of vehicle defects is the same for every minute that the vehicle drives.
- Vehicle defects and breakdowns can only occur during a driving activity of the vehicle.

Triggers & rescheduling

- When a trigger for rescheduling occurs, the simulation is temporarily stopped to calculate rescheduling options. When finished with the rescheduling calculations, the simulation is continued with the new routes as determined in the rescheduling.
- The location of vehicles when a rescheduling operation is triggered are determined as follows (more information on the vehicle states is included later in this section):
    - At the stop when the state of the vehicle is “service”.

    - At a distance that is equal to the time driven by the vehicle divided by the total driving time for the path when the state of the vehicle is “drive“.
    - At the break location (always a customer) when the state of the vehicle is “break“.
    - At the wait location (always a customer) when the state of the vehicle is “wait“.
    - At the vehicle breakdown location when the state of the vehicle is “breakdown”.
    - At the vehicle defect location when the state of the vehicle is “defect”.
- Transferring goods takes a fixed time of 3 minutes plus 0.5 minutes for each unit of demand that is transferred.
- Vehicles can park everywhere. When a transfer location between two stops is used, it is assumed that the vehicles are always able to park in this location.
- When a route is rescheduled with an inter-route transfer, the vehicle cannot be rescheduled anymore during the rest of the simulation.
- When a route is rescheduled with an intra-route switch, the vehicle cannot be rescheduled for an hour.
- Vehicles that are involved in another rescheduling operation are occupied and cannot help out other vehicles that need rescheduling.
- We set a high penalty (1,000,000 minutes) on time window violations for new customers. This way, new customer time window violations are not impossible, but options without time window violations for new customers will always be preferred.
- We use the same penalty function for new customers as for recurring customers in the experiments of the objective function phase where there is no new customer distinction.

### 5.1.2 Objects & attributes

We define objects and object attributes for our simulation model here. Figure 17 presents the objects and object relations of the objects used in our model. If a relation between two objects exists, we use an arrow for this relation with a given multiplicity for the relation. We use the following objects in our simulation model:

- Plan: this is the planned collection of vehicles, customers, the sequence of the customers (a route), and orders. Plans are made by the case study retailer and separate planning instances are available for each morning (PO) and afternoon (PA) shift. During the simulation, routes might be altered by rescheduling, but the collection of orders remains intact because of the assumption that no new demand arrives dynamically over time.
- Vehicle: all vehicles are modelled as separate objects. The vehicles have several attributes, all shown in Figure 17. A vehicle always has a single vehicle id and “ritid” (id for the route). A route_type attribute represents the type of route. The list of stops is an ordered collection of locations that are to be visited by the vehicle driving the route. The list of stops can be altered during the simulation. Order ids in the vehicle that belong to a stop are removed from the vehicle when the stop is visited. When no more order ids are in the vehicle, the vehicle is empty. State indicates what the vehicle is doing at that moment in the simulation. Table 3 displays the possible states of a vehicle. The state of the vehicle is updated for each simulation event (see the next part of this section for more explanation). A vehicle has an is_rescheduled attribute to indicate whether or not the vehicle has just been rescheduled. If this is set to ‘True’ the vehicle is not available for rescheduling. When a vehicle is involved in a transfer, this attribute remains ‘True’ for the rest of its route. When an en-route order switch is done for the vehicle, a timeout is set for when the is_rescheduled attribute is set back to its default value: ‘False’, meaning the vehicle can be triggered for rescheduling again.

- Stop: in most cases, stops are customers that placed an order with a time window (startTW to endTW). The depot is also included as a stop at the start and end of the route. For rescheduling purposes, two additional stop types are introduced: transferpoints and current locations. At transferpoints, goods are transferred from one vehicle to another. A transferpoint can have the same location as a stop, but it is modelled as a separate stop. The current location stop type is used when a vehicle is driving. Another stop type is the dummy customer. Backup vehicles drive towards a dummy customer to wait here until they are possibly needed in rescheduling actions. They dummy customers can also be located at a depot. All possible stop types are summarized in Table 4. Next to a stop type, each stop has a name and location (x, y). All stops also have an order id and a Boolean new_cust attribute. Remaining attributes of the Stop-object represent arrival (planned and realized), departure (planned and realized), wait and break times.
- Order: the object order is a given number of packages that belongs to a customer's order_id. Order is modelled as a separate object to be able to also refer to orders from the vehicle object.

*Table 3: possible vehicle states during a simulation run.*

| **Vehicle state** | **Description** |
|---|---|
| Loading | Vehicle is loading orders at the depot. |
| Driving | Vehicle drives towards one of the stops on his route. |
| Servicing | Vehicle is busy servicing a stop (delivery of goods). |
| Breaking | Vehicle takes a break in the delivery process. |
| Waiting | Vehicle waits till the start of the time window to start service. |
| Breakdown | Vehicle is broken down and cannot continue its route. |
| Defect | Vehicle receives minor roadside assistance to solve technical problems. |
| Drive_empty | Vehicle has delivered all orders on its route and drives back empty to its depot. |
| Backup | Vehicle waits as a backup vehicle at a given location. |
| Finished | Vehicle is finished with its entire route and it is no longer available for the rest of the simulation process. |

*Table 4: possible stop types within the simulation.*

| **Stop type** | **Description** |
|---|---|
| Current_location | The current location of a vehicle, only relevant when rescheduling. |
| Customer | A customer that ordered goods at the retailer. Goal is to visit all customers within specified time windows. |
| Transferpoint | A location where goods are transferred from one vehicle to another. A transferpoint is part of two different routes, so it is visited by two vehicles. |
| Dummy_customer | A dummy customer that is used for backup vehicles. They drive towards this dummy customer a 'service' this customer. Because it is a dummy customer, the service time can be interrupted in this case. |
| Depot | The start and end location of a vehicle. |

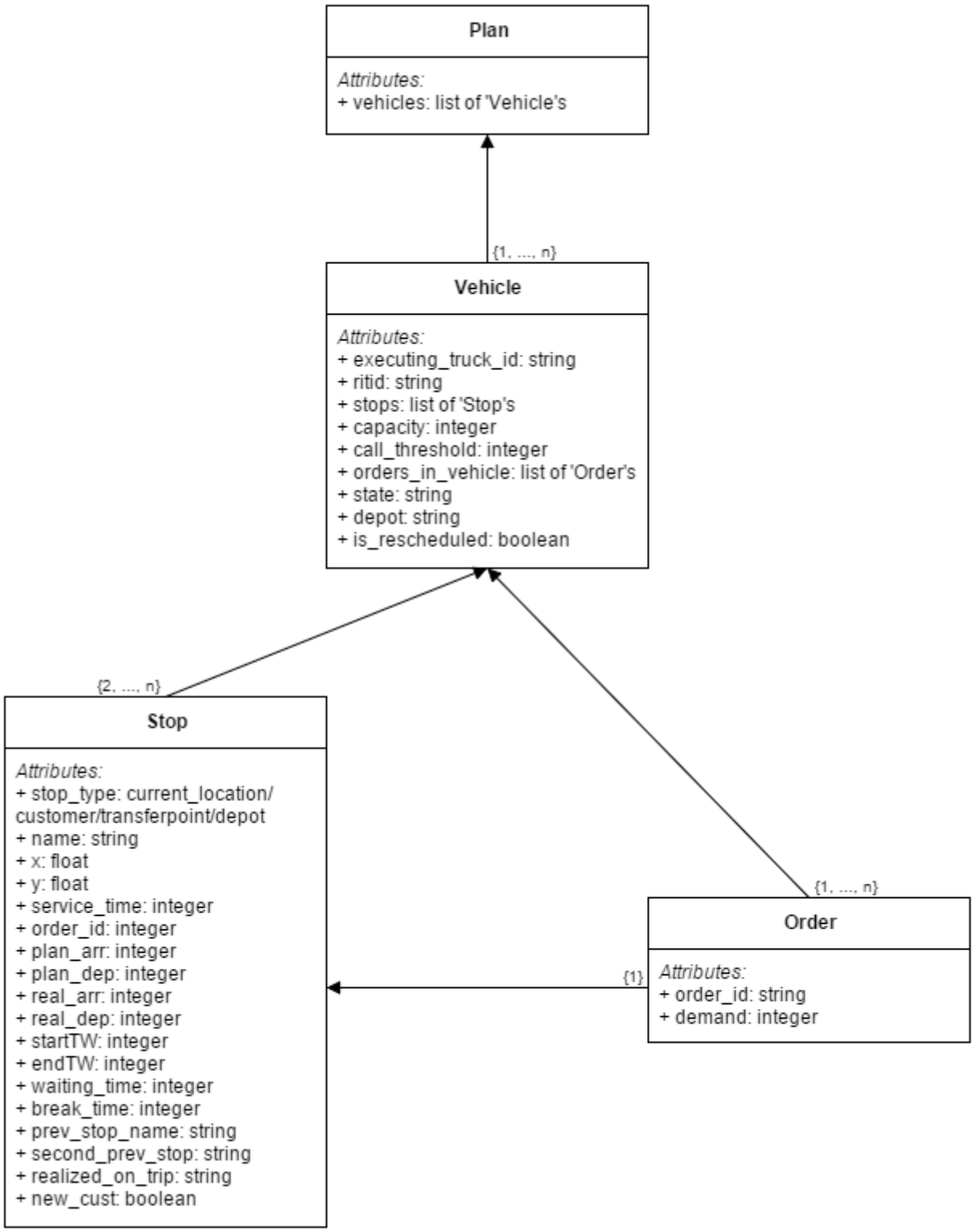


*Figure 17: object relations, types and attributes of the model implementation.*

### 5.1.3 Events

In our simulation for en-route rescheduling we define events that cause the state of our system to change. Because we apply the three phase approach, Booked (B) events and Conditional (C) events are set for in our event list. Table 5 displays the events. An event is finished at the same moment that the next event starts. Figure 18 displays how a simulation replication steps through the event list in our simulation model. Here, the 24 shifts in our test set are a single replication. When a trigger from either the benchmark heuristic or the Tabu Search rescheduling algorithm is triggered during one of the simulation runs, the simulation is paused and the rescheduling actions are performed.

*Table 5: event list for rescheduling test simulation.*

| Event name | Belongs to [Object] | Event type | Event description |
|---|---|---|---|
| Start of shift | Plan | C | The earliest time a vehicle finished loading at the depot, the shift starts. |
| Load | Vehicle | B | We only know the end of the loading time in our simulation. Departure delays can occur during the loading time. Realized delays are used. When no realization of the departure is known, the planned departure time is used. |
| Drive | Vehicle | B | A vehicle starts driving towards its next stop upon finishing a service time at a previous customer. The drive duration is based on a simulated driving time. |
| Break | Vehicle | B | A vehicle starts a break according to the planned break time after finishing the previous event. The break is held at the location of a stop before the start of a service. |
| Wait | Vehicle | C | A vehicle starts a waiting events when it finishes the driving event more than 15 minutes earlier than the start of a time window. The vehicle waits with the start of a service until it is allowed to start. |
| Service | Vehicle | B | A vehicle starts the service of a customer at the moment of arrival (if no break or waiting time is relevant). The service time is based on a realized service time. When no realized service time is available the planned service time is used. |
| Reschedule | Vehicle, Plan | C | Triggers are checked and if a trigger is activated, the vehicle and potential helper vehicles from the search space selection function in the plan are rescheduled. A switch is instantly effective. A transfer is added as a stop to the relevant routes. |
| Transfer | Vehicle | B | A vehicle starts to transfer after deciding on the transfer in a rescheduling calculation. The transfer |

| | | | |
|---|---|---|---|
| | | | time needed is based on the number of demand units that have to be transferred. Orders are moved from one vehicle to another. |
| Breakdown | Vehicle | C | When the state of a vehicle is “driving” and the current simulation time is larger than the breakdown time of the vehicle, a breakdown starts and the vehicle cannot continue its route. |
| Defect | Vehicle | C | When the state of a vehicle is “driving” and the current simulation time is larger than the defect time of the vehicle, a defect starts and an hour is needed to solve this vehicle defect. |
| Arrive at depot | Vehicle | B | All customers are visited and the vehicle is back at the depot. The vehicle is now no longer available for rescheduling. |
| End of shift | Plan | C | If all vehicles are finished with their route, we set the end of the shift. We then continue with the next shift. |

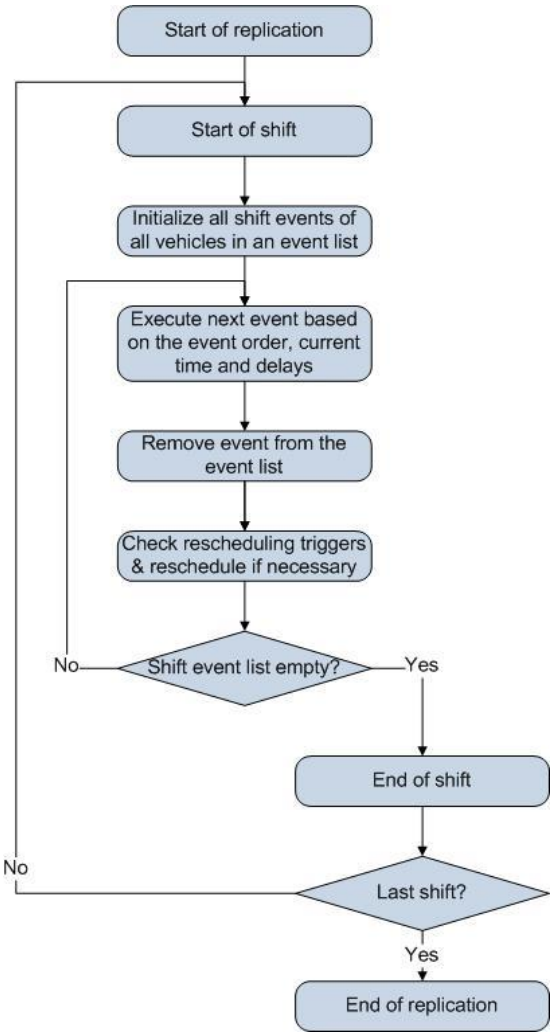


*Figure 18: event handling in simulation runs.*

### 5.1.4 Model input

We describe the simulation data, number of replications, and experimental factors here.

#### *5.1.4.1 Simulation data*

We use actual planning data of our retailer case study as the planned instances for our simulation runs. We use planning instances of all shifts between September 28th and October 11th for our simulation runs. We only use two weeks of the total available dataset because of the total runtime needed for simulation runs. We selected the last two weeks of the available dataset because the registration of delivery times in the DCT was the best for these weeks (see Appendix A).

We use common random numbers for tests of the different scenarios and methods, meaning that all events with variability have the same characteristics for the different scenarios and methods that are compared. Common random numbers are applied for each shift and replication over the different experiments. The events that are based on variability are: vehicle defects, vehicle breakdowns, driving events (travel times), and operator selection within the allowed operator types. All distributions used for the travel times in the simulation replications are based on the distributions found in Chapter 3 and Appendix B. We use the defect and breakdown probabilities as defined at the start of this section. We do our operator selection using uniform distributions.

#### *5.1.4.2 Number of replications*

The time windows are already specified in the plan we retrieve from the database of the DCT in our simulation replications (runs). We can assume that the demand or time windows cannot change anymore after the plan is uploaded in the DCT. This is not allowed in the real-world case as well.

We deal with a terminating solution, meaning that the simulation will stop once no more events are scheduled. Therefore, long simulation runs are not an option, and we choose to do multiple replications for each of the planning instances. A replication is a single run of a simulation model that uses specified streams of random numbers, which in turn cause a specific sequence of random events. By performing multiple replications and taking the mean of the results, a better estimate of model performance is gained.

Variables that are based on variability in our simulation model are: the number of vehicle defects, the number of vehicle breakdowns, travel times, and the selection of an operator within the allowed operator types. Some replications will include defects and breakdowns, but we select the travel times to determine the number of replications because we sample travel times for all paths from the distributions we defined in all experimentes.

We determine a number of replications for our simulation experiments using the confidence interval method described in Robinson (2004). This method keeps track of all travel times in the simulation to see how well the mean of a variable is being estimated. The narrower the interval the more accurate the estimate is deemed to be. A confidence interval can be determined using Equation 11. We define *CI* as the confidence interval, *t* as the value of the t-distribution based on *n-1* degrees of freedom and $\alpha$ as confidence value (we use a 99% confidence interval). Furthermore, we define *S* as the standard deviation of the output data of the replications, $\bar{X}$ as the average travel time over a replication, and *n* as the number of replications.

$$CI = t_{n-1,\alpha/2} * \frac{S}{\sqrt{n}} \tag{11}$$

We accept deviation up to 1% for our simulation runs because one run is long, meaning that many travel times are sampled in each of the runs. Table 6 shows the results of the confidence interval method, where a percentage deviation is eventually determined on the lower- and upper intervals. Based on the results we find, we determine that we use 3 runs for each of our experiments.

*Table 6: Number of runs determination using the confidence interval method on travel times within the simulation model.*

| Run | Mean travel time [min] | Cumulative mean [min] | Standard deviation | Lower interval | Upper interval | Percentage deviation |
|---|---|---|---|---|---|---|
| **1** | 9.1162 | 9.1162 | n/a | n/a | n/a | n/a |
| **2** | 9.1223 | 9.1193 | 0.0044 | 8.9230 | 9.3155 | 2.15% |
| **3** | 9.1175 | 9.1187 | 0.0032 | 9.1001 | 9.1373 | 0.20% |

#### *5.1.4.3 Experimental factors*

The experimental factors are the input variables that are varied over the different experiments. As explained before, we test two types of methods: a benchmark heuristic and our Tabu Search Rescheduling (TSR) method. Table 7 presents the experimental factors used in our simulation runs. If we would experiment with all possible combinations, this would result in a lot of experiments. Because it is difficult and time-consuming to analyze too many experimental results, we propose a simulation setup that consists of the following phases:

1. *Calibration phase* aims to find settings that ensure realistic and feasible solutions for rescheduling;
2. *Trigger test phase* aims to find the influence of different types of trigger sets for the benchmark and Tabu Search rescheduling methods;
3. *Objective function phase* aims to find good settings for the objective function in Tabu Search rescheduling for our use case;
4. *Operator selection phase* aims to find good 'local' operator settings for our use case;
5. *Managerial insights phase* aims to give additional insights for management purposes. The experiments are defined based on findings in other experiments. We define the managerial insights experiments in Section 6.5.

The first phase fixes some parameters for use in the rest of the experiments. We consider experiment 1 and 2 as our base experiments for the benchmark heuristic and TSR method respectively. Phases 2, 3, and 4 are used to see the influence of triggers, objective function variations, and allowed operators for rescheduling respectively. In the operator selection phase we distinguish between intra- and inter-route rescheduling, because the 2*-operator can only be used in inter-route rescheduling. We include a final phase to show some additional insights on the results of our rescheduling method for management purposes. Table 8 shows the results of different combinations of experimental settings in the experiment settings for our simulation model.

*Table 7: Experimental factors.*

| Simulation phase | Parameter | Values |
|---|---|---|
| 1) Calibration | Method | Benchmark Rescheduling Heuristic (BRH)<br>Tabu Search Rescheduling (TSR) |
| 2) Trigger tests | Triggers | Benchmark triggers<br>TS triggers |
| | Robustness trigger in minutes | 5 minutes<br>10 minutes |
| 3) Objective function | TW violation penalty | Linear penalty function |

| | | |
|---|---|---|
| | | Quadratic penalty function<br>5 minutes 'grace' (HTW)<br>15 minutes 'grace' (HTW) |
| | Robustness bonus | No bonus<br>Robustness bonus function |
| | Whether or not we distinguish new customers in our objective function | Yes<br>No |
| 4) Operator selection | The use of different types of exchange/2-opt*/Or-opt operators in the 'local' Tabu Search part of the rescheduling algorithm | 33% exchange, 33% 2-opt*, 33% Or-opt<br>100% Or-opt<br>100% exchange<br>50% exchange, 50% Or-opt<br>50% 2-opt*, 50% exchange<br>50% 2-opt*, 50% Or-opt |
| 5) Managerial insights | Additional analyses that are interesting for management. | To be defined in Chapter 6. |

### 5.1.5 Model output

We compare the performance of different methods and settings to determine the solution quality of different TSR settings. The performance measures we use to compare the different solutions are listed below. We present total numbers of the two weeks of simulation data, where the average values are used per shift. This way we limit the amount of output we analyze, while we are able to state how much we can save over a time period of two weeks.

- New customers late [#]: a measure for the number of new customers that receive a late delivery for customers that place a first time order at the retailer;
- Recurring customers late [#]: a measure for the number of recurring customers that receive a late delivery for customers that have placed order at the retailer before;
- Violation time [min]: the total time that vehicles arrived after the end of the time window summarized over all late deliveries;
- Total distance [km]: the distance that all vehicles together travel to service all customers;
- Switches made [#]: the number of times a rescheduling calculation results in an intra-route switch of customers;
- Transfers made [#]: the number of times a rescheduling calculation results in an inter-route transfer of customers:
- CPU time for rescheduling calculations [s]: the time in seconds we need for our TSR method.

Additional output that we do not show in our main report, but only in our more detailed representation of the results in Appendix D, is listed below.

- Vehicles [#]: total number of vehicles we use to service all customers;
- New cust. [#]: total number of new customers in all shifts;
- Rec. cust. [#]: total number of recurring customers in all shifts;
- Total time [min]: the total time needed for all vehicles to service all customers (includes break, travel, service and waiting times);
- Breakdowns [#]: the average number of breakdowns for the experiment;
- Defects [#]: the average number of defects for the experiment;
- No changes [#]: the number of times that TSR calculations did not find a better option than to continue as planned before.

*Table 8: Experiment settings for the simulation model.*

| Exp | Method | Triggers | Robustness trigger | TW violation penalty | Robustness bonus | New cust. distinction | intra-route operators | Inter-route operators |
|---|---|---|---|---|---|---|---|---|
| **1** | BRH | BRH | - | - | - | - | - | - |
| **2** | TSR | TSR | No | Linear | No | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **3** | TSR | TSR | Yes | Linear | No | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **4** | BRH | TSR | Yes | - | - | - | - | - |
| **5** | BRH | TSR | No | - | - | - | - | - |
| **6** | TSR | BRH | - | Linear | No | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **7** | TSR | TSR | No | Quadratic | No | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **8** | TSR | TSR | No | 5-min HTW | No | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **9** | TSR | TSR | No | 15-min HTW | No | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **10** | TSR | TSR | No | Linear | Yes | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **11** | TSR | TSR | No | Quadratic | Yes | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **12** | TSR | TSR | No | 5-min HTW | Yes | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **13** | TSR | TSR | No | 15-min HTW | Yes | Yes | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **14** | TSR | TSR | No | Linear | No | No | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **15** | TSR | TSR | No | Quadratic | No | No | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **16** | TSR | TSR | No | 5-min HTW | No | No | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **17** | TSR | TSR | No | 15-min HTW | No | No | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **18** | TSR | TSR | No | Linear | Yes | No | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **19** | TSR | TSR | No | Quadratic | Yes | No | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **20** | TSR | TSR | No | 5-min HTW | Yes | No | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **21** | TSR | TSR | No | 15-min HTW | Yes | No | 50%EX, 50%OR | 33%2*, 33%EX, 33%OR |
| **22** | TSR | TSR | No | Linear | No | Yes | 100%OR | 100%OR |
| **23** | TSR | TSR | No | Linear | No | Yes | 100%EX | 100%EX |
| **24** | TSR | TSR | No | Linear | No | Yes | 50%EX, 50%OR | 50%EX, 50%OR |
| **25** | TSR | TSR | No | Linear | No | Yes | 50%EX, 50%OR | 50%2*, 50%EX |
| **26** | TSR | TSR | No | Linear | No | Yes | 50%EX, 50%OR | 50%2*, 50%OR |

## 5.2 MODEL IMPLEMENTATION

This section describes how the rescheduling methods are implemented in the simulation model. We first describe the benchmark rescheduling heuristic, which mimics the control room rescheduling actions. After this, we describe the Tabu Search Rescheduling (TSR) method implementation.

### 5.2.1 Benchmark heuristic

Vehicle drivers are obliged to call the control room about their lateness when they are either stranded (e.g., due to a vehicle defect or breakdown), or when they are 20 minutes or more behind schedule. This is, in the benchmark heuristic, the moment that the control room team is triggered to start rescheduling. We assume that stranded vehicles are instant triggers for the control room team. They state that the 20 minutes lateness call is correctly done by most drivers. Still, some variation can be assumed for the time of calling. To mimic drivers calling about their lateness we use a uniform distribution with 20 minutes late and 30 minutes late as its boundaries. Each vehicle has a randomly assigned threshold value from this uniform distribution for the moment it will call the control room. When the lateness exceeds this threshold we simulate a call by the driver in our benchmark heuristic. The control room team starts with checking if there are stops where the current delay will ensure late delivery based on delay and time windows. If not, the simulation continues without setting a timeout till the next trigger. If a problem exists for one or more stops, these stops are remembered in the next steps of the benchmark heuristic.

The control room team first tries to solve the problem for the problem stops by moving stops forward in the route sequence, so that they will receive their order earlier. When this solves at least a part of the problem without creating another problem in the route, the stop is moved in the order. They check if there still is a problem after this. If this is the case, they continue to look for a transfer. The control room team checks for empty helper vehicles in a range of 15 kilometers Euclidian distance. This is a large range, but they do this because they only work with empty helper vehicles so no other deliveries are endangered by reschedule actions. If a helper vehicle is found, the empty vehicle drives towards one of the next stops of the disrupted vehicle to transfer customer orders.

In case of a breakdown, the situation is somewhat different than when lateness triggers rescheduling. When a breakdown occurs, the nearest empty vehicle is used as helper to pick up the customer orders from the breakdown location of the disrupted vehicle. No maximum distance of the helper is used in this case.

When no switch or transfer identified by the benchmark heuristic can ensure in-time delivery for all customers, the customers that will receive a late delivery are informed about the lateness by the customer service department of the retailer.

### 5.2.2 Tabu Search rescheduling

For the Tabu Search rescheduling, we fully described the methodology in Chapter 4. However, we used some simplifications and generalizations for our triggers when implementing the method in our simulation model. The triggers are implemented in the model setup in the following way:

- **ETA/Robustness trigger**: expected violation of time windows based on current lateness of a vehicle and expected travel times. A rescheduling operation is triggered here when the expected lateness of a vehicle is larger than the smallest difference between expected start time at a customer and the end of the time window. When using a robustness trigger, not the end of the time window, but the end of the time window minus some additional robustness is used to earlier signal a need for rescheduling. The use of the robustness trigger is varied in

different experiments. We assume that no other (faster) paths are available for a vehicle when its ETA trigger is triggered.

- **Vehicle defect trigger:** vehicle defects are problems with a vehicle that can be solved with some minor roadside assistance. When such a problem occurs, this trigger is instantly activated and a rescheduling is initiated.
- **Vehicle breakdown trigger:** we consider vehicle breakdowns as a vehicle that is not able to continue its route. When such a problem occurs, this trigger is instantly activated and a rescheduling is initiated.

Other triggers we mentioned are accident and event triggers. These are hard to include in the simulation setup however, and no data on the actual disruptions caused by accidents and events is available. We therefore omit these trigger types from our simulation experiments.

## 5.3 MODEL VERIFICATION & VALIDATION

This section describes the model verification and validation steps subsequently. The verification step aims to verify if the simulation works as intended. The validation step determines if the simulation model is an accurate representation of the real system.

### 5.3.1 Verification

The verification process is concerned with determining if a simulation program works as intended (Law, 2007). Verification is done as an iterative process throughout the model construction phase. Every time new objects are added to the model, the correct implementation is checked by printing all attribute changes of objects on events. When the attribute changes are correct with respect to the conceptual model for each type of event, new objects and functions are added to the model, verifying them as well.

We also verified the correctness of the simulated travel times, defects and breakdowns here. Table 9 shows both the historical mean and variance and the mean and variance of the model output. We find that the differences are small enough to verify that our model input is correct. We do see lower model output mean values for the travel times, but the differences are small. The standard deviations are similar as well for the historical and model output variables. The number of runs and the implementation of defects and breakdowns seem to ensure realistic percentages for these verification numbers in the simulation output.

*Table 9: Verification of simulation model output.*

| Name | Historical mean [min] | Historical standard dev. [min] | Model output mean [min] | Model output standard dev. [min] |
|---|---|---|---|---|
| Travel times in distance class 0-0.5 km | 2.94 | 2.00 | 3.15 | 2.92 |
| Travel times in distance class 0.5-1.0 km | 5.92 | 2.61 | 5.76 | 2.70 |
| Travel times in distance class 1.0-1.5 km | 7.63 | 3.04 | 7.34 | 2.80 |
| Travel times in distance class 1.5-2.0 km | 9.04 | 3.27 | 8.66 | 2.95 |
| Travel times in distance class 2.0-2.5 km | 10.27 | 3.45 | 9.89 | 3.17 |
| Travel times in distance class 2.5-3.0 km | 11.33 | 3.79 | 10.80 | 3.38 |
| Travel times in distance class 3.0-4.0 km | 12.73 | 4.28 | 12.13 | 3.94 |
| Travel times in distance class 4.0-5.0 km | 14.31 | 4.59 | 13.63 | 4.22 |
| Travel times in distance class 5.0-6.0 km | 15.96 | 4.92 | 15.18 | 4.40 |
| Travel times in distance class 6.0-7.0 km | 17.63 | 5.49 | 16.81 | 5.11 |
| Travel times in distance class 7.0-8.0 km | 18.96 | 5.80 | 18.09 | 5.25 |

| | | | | |
|---|---|---|---|---|
| Travel times in distance class 8.0-10.0 km | 20.79 | 6.00 | 19.92 | 5.66 |
| Travel times in distance class 10.0-12.5 km | 22.94 | 6.05 | 22.06 | 5.72 |
| Travel times in distance class 12.5-15.0 km | 25.37 | 6.57 | 24.45 | 6.27 |
| Travel times in distance class 15.0-20.0 km | 29.62 | 7.75 | 28.90 | 7.96 |
| Travel times in distance class 20.0-25.0 km | 34.17 | 8.45 | 33.02 | 8.39 |
| Travel times in distance class 25.0 km < | 41.03 | 11.13 | 41.99 | 12.69 |
| Vehicle Breakdowns | 0.06% | - | 0.05% | - |
| Vehicle Defects | 0.60% | - | 0.62% | - |

### 5.3.2 Validation

The validation process determines whether the simulation model is an accurate representation of the system, for the objectives of the study (Law, 2007). We check if the overall model provides a sufficiently accurate representation of the real world system here. We check if the results of running an experiment with common random numbers gives similar results to the actual delivery times and durations. We validate on the model output of experiment 1: the benchmark heuristic base experiment. This experiment aims to model the current method of rescheduling. We compare the output of this experiment with the realized data from the DCT database.

Figure 19 shows that the distribution of time window violation times in our benchmark experiment is similar to the one found in the realization data. Largest differences are found in the category 0-5 minutes delay – where the realization data has a larger percentage – and the categories with the most extreme delays – where the benchmark experiment has a larger percentage. This indicates that we overestimate the number of extreme travel delays in our experiments.

Figure 20 and Figure 21 display the validation results on numbers of late deliveries and total time needed for delivery respectively. We use percentage deviations of our simulation results from the realizion data in the DCT to preserve data confidentiality. We include the realization data from the DCT itself in Appendix C. We calculate the percentage deviation using: *(simulation – realization) / realization*. Figure 20 shows that we overestimate the total number of late deliveries in our simulation experiments, where we used a total number of late deliveries for recurring and new customers. The general pattern is similar to the realization data, but we do not find a fit of our simulation results. We also see this in the fluctuations of our percentage deviation. The total number of late deliveries is too high for most days in the simulation data. The overestimation of the number of late deliveries might be explained by the fact that our benchmark heuristic only looks at solving the number of 'problems' on routes. The control room team probably takes some measure of problem severity into account that we did not model in our benchmark heuristic.

Figure 21 shows that the total time for all vehicles and deliveries in the benchmark experiment is very close to the realized total time. We find most deviations within a margin of 5%, with maximum deviations of 8%. We see that morning shifts often underestimate the total time, while afternoon shifts often overestimate the total time. This indicates that there is a distinction between morning and afternoon travel times. Still, the deviations are small enough to say that the travel times and fit of total time are good enough for our simulation model.

We overestimate the total number of late deliveries in our simulation model. We conclude that we model reality close enough to make a fair comparison between experimental settings and different numbers of late deliveries. The total time fits the real situation well and the numbers of late deliveries are comparable to the real situation. The model enables us to show the results of our TSR method well, but the precise benefits of the method might differ slightly when actually implemented in the DCT.

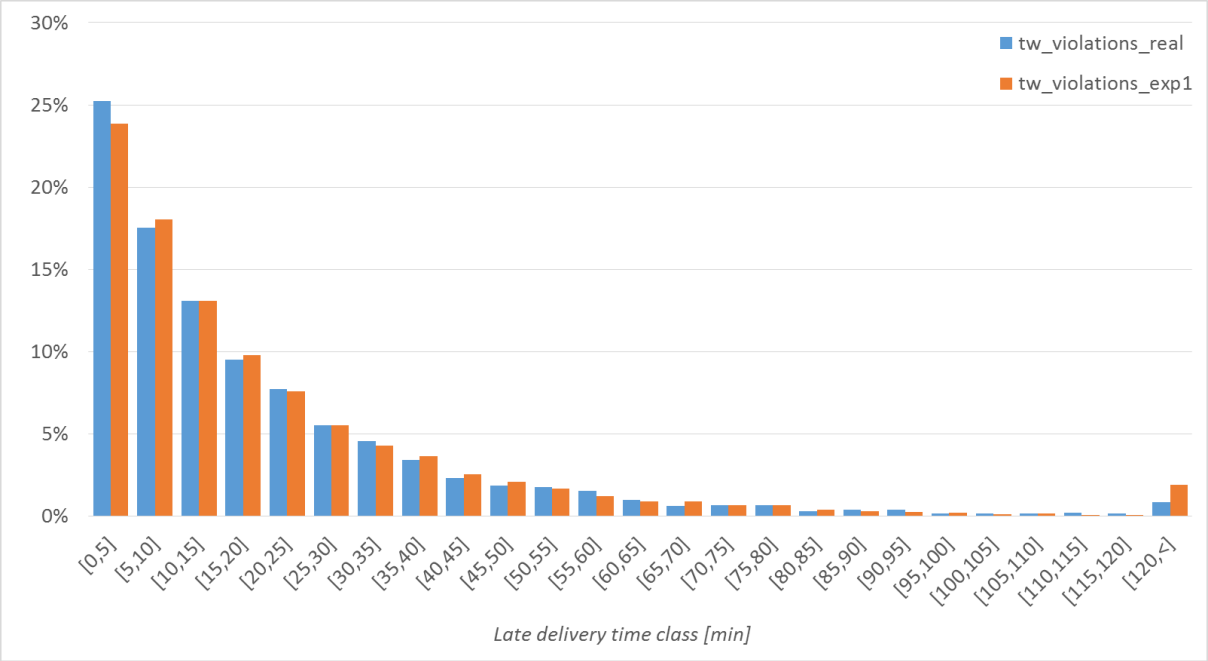


*Figure 19: histogram of time window violations according to the realization data and experiment 1.*

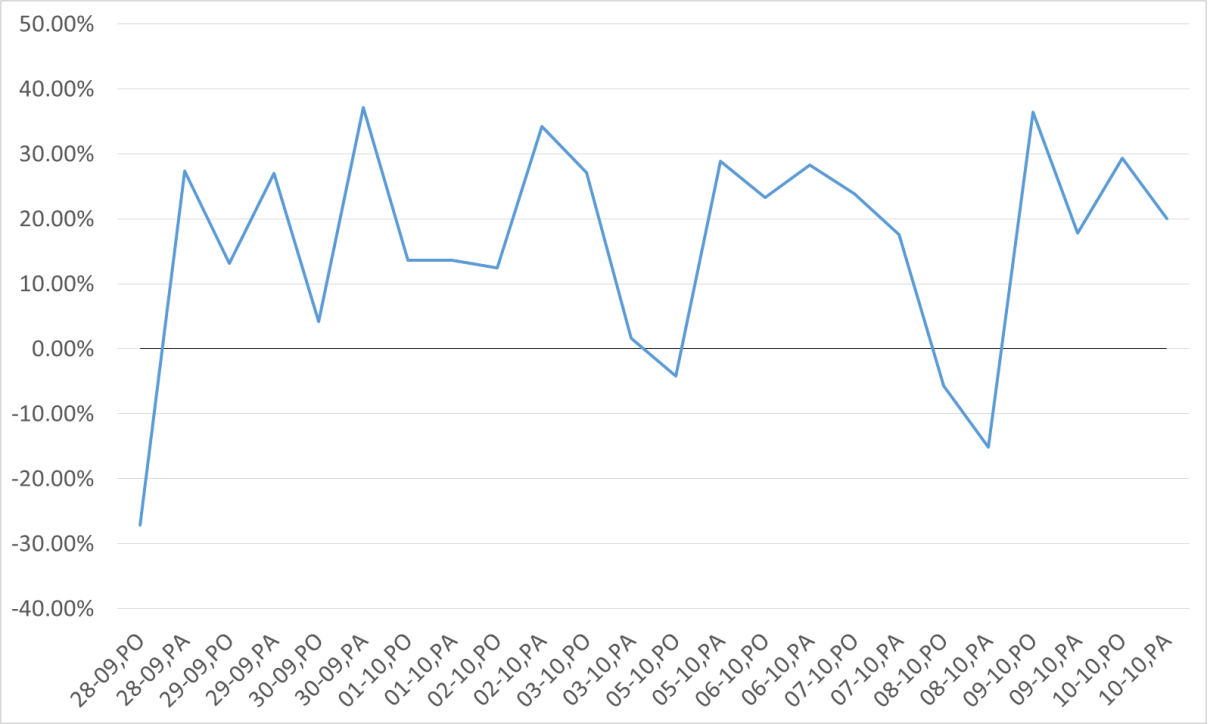


*Figure 20: percentage deviation of the simulated total number of late deliveries (number of recurring + new customers with late delivery) from the realized late deliveries per shift in the test set*

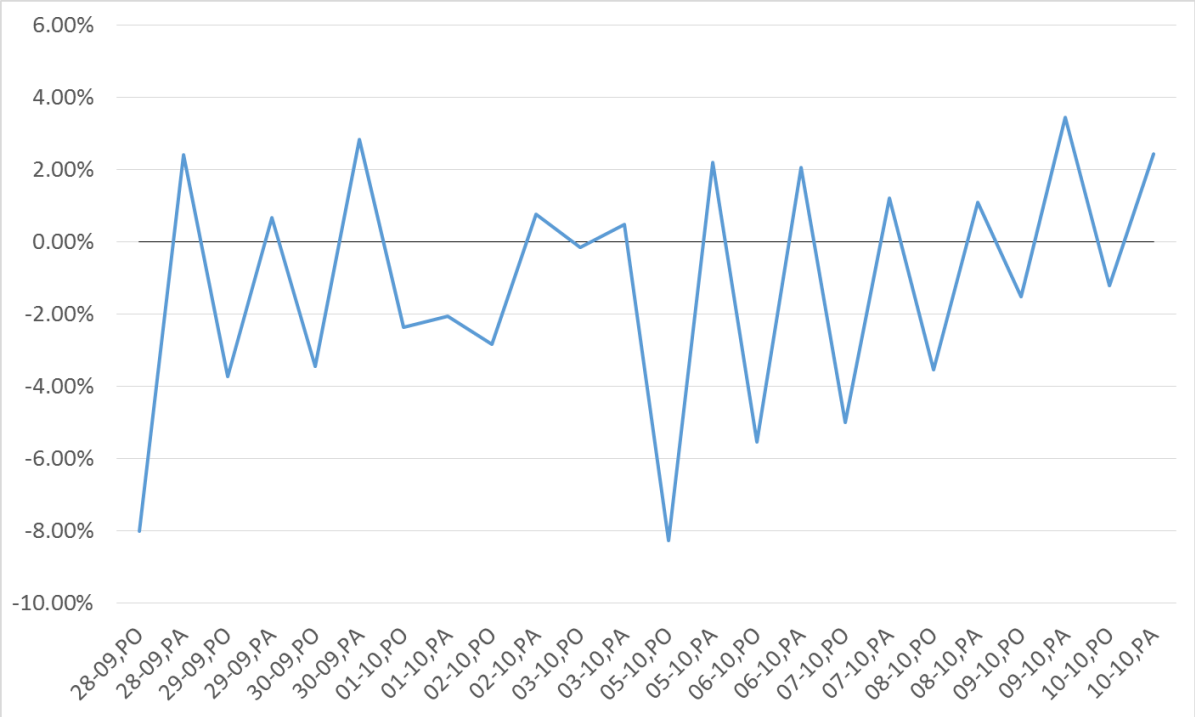


*Figure 21: percentage deviation of simulated total time (including all service, travel, wait, break times) needed for all deliveries from the realized total time needed for all deliveries per shift in the test set.*

## 5.4 SIMULATION MODEL CONCLUSION

To test the TSR method settings and performance, we develop a simulation model. We decide to run the simulation model on two weeks of planning instances of the case study retailer, which include morning (PO) and afternoon (PA) shifts. We model all activities of the delivery vehicles, and we verify and validate the simulation model on realizations from the DCT databases. We verify our simulation model, but we find that we overestimate the number of late deliveries. We conclude that we model reality close enough to make a fair comparison between experimental settings. The total time fits the real situation well and the numbers of late deliveries are comparable to the real situation. The model enables us to show the results of our TSR method well, but the benefits of the method might differ slightly when actually implementing it in the DCT.

# 6 RESULTS

In this chapter, we present the results of our simulation study on en-route rescheduling of a home delivery operation. We present the results of the calibration phase in Section 6.1, where we aim to find good model parameters for our Tabu Search Rescheduling (TSR) method that we can fix during the simulation experiments. We describe the results of the trigger test, objective function and operator selection phases in Section 6.2, 6.3 and 6.4 subsequently, where we aim to find the best settings for our TSR method. In Section 6.5, we describe managerial insights on on-time departure, backup vehicle locations, combining settings from the other phases, knowledge of future travel times, initial planning quality and phased TSR method implementation. We describe result conclusions in Section 6.6. We present totals over the simulation time period to get a general insight into the results of the simulation experiments. We include the elaborate simulation results in Appendix D.

## 6.1 CALIBRATION

We aim to find good model parameters for our TSR method that we can fix during the simulation experiments. We display these parameters in Table 10. We use experiment 2 as defined in the previous chapter, with the basic setting for triggers, objective function parameters and operators, to calibrate these values. Calibration goals are the following:

- Number of new and recurring customers that receive deliveries should be equal to the planned number of customers;
- One simulation run should not require more than 2.5 minutes per shift in the experiment runs (meaning that a single replication over all 24 shifts takes no more than 1 hour of simulation time) to ensure we can run all simulation runs and experiments in reasonable time;
- The system should not be too nervous, i.e., triggers should not go off all the time;
- Number of vehicles in the search space of a solution should be small (i.e., around 5 vehicles);
- The plans of the Tabu Search Rescheduling algorithm should “make sense”. This means we analyze the plans visually based on time windows and geographic location. When the results are not logical, more calibration of the model and algorithm is needed;

We first run experiment 2 with the settings as displayed for step 1 in Table 10. The number of new and recurring customers that received deliveries are equal to the number of planned customers in both categories, meaning that no customer deliveries are “lost” in the TSR method and simulation process. We find the rescheduling calculations to be running far too long, resulting in too large running times for a single run (up to 3 hours). We therefore tune the stopping criteria of the Tabu Search: the maximum number of iterations without improvement and maximum allowed CPU time per rescheduling operation to be lower (both intra-route and inter-route). We differentiate between intra-route and inter-route here, because the number of options is generally much lower for the intra-route search. Furthermore, we check if the system is not too nervous in rescheduling, meaning that it would keep on rescheduling routes that have just been rescheduled. This is the case, so we want to limit the total number of rescheduling calculations. We therefore prohibit vehicles to be in another rescheduling action when they were already involved in a transfer. For the switch moves, a timeout of an hour is used until they can be triggered again. Next to this, we find that the settings mentioned for step 2 satisfy the time condition we set.

Next, we run the simulation experiment with the settings for step 2 in Table 10. We find that solutions are visited more than once in the search process. We therefore make our Tabu lists longer to ensure

that solutions are not visited more than once in the search. We find that a Tabu list of length 50 ensures that this is the case.

In step 3 of the calibration phase we find that the robustness bonus is too large, ensuring that solutions with late deliveries for certain customers are too much compensated by the robustness of others. Therefore, we decrease the maximum robustness bonus to 3 minutes, ensuring that solutions with expected lateness of delivery are less favorable from solutions without lateness.

From our first few experiments we find that transfer penalties in the objective function would be a bit overkill for transfer (un)attractiveness in our case study. The additional driving time and time that is required to transfer orders makes solutions that include transfers unattractive enough to not use transfer penalties.

The calibration steps are summarized in Table 10. We show the parameter settings at the start of the calibration, the parameter settings at the two steps in between, and the final parameter settings that follow from the calibration. The no improvement and CPU time parameters are the stopping criteria of the Tabu Search algorithms, with separate parameters for inter-route and intra-route moves. The Tabu list length parameters define the maximum length of the Tabu list containing already visited solutions. The maximum robustness bonus parameter defines the maximum bonus given in the objective function of the TSR method on a customer level.

*Table 10: Calibration steps and values for the different parameters and calibration goals.*

| **Calibration parameter** | **Step 1** | **Step 2** | **Step 3** | **Final settings** |
|---|---|---|---|---|
| No improvement for $I_{inter}$ consecutive iterations | 200 | 50 | 50 | 50 |
| CPU time of inter-route algorithm > $T_{inter}$ seconds | 30 | 10 | 10 | 10 |
| No improvement for $I_{intra}$ consecutive iterations | 200 | 30 | 30 | 30 |
| CPU time of intra-route algorithm > $T_{intra}$ seconds | 30 | 5 | 5 | 5 |
| Tabu list length inter-route | 20 | 20 | 50 | 50 |
| Tabu list length intra-route | 20 | 20 | 50 | 50 |
| Maximum robustness bonus in minutes | 5 | 5 | 5 | 3 |

## 6.2 TRIGGERS

This section presents the results of experiments 1 to 6 to find the effectiveness of triggers on both the benchmark method and the TSR method. Experiment 1 and 2 are the base experiments for the benchmark heuristic and TSR respectively. Experiment 3 adds a robustness trigger to the TSR, where triggers are activated when a vehicle is expected to arrive in the last five minutes of a time window. Experiment 4 and 5 use the TSR triggers in combination with the benchmark heuristic, with a robustness trigger and no robustness triggers respectively. Experiment 6 combines the benchmark triggers with the TSR rescheduling algorithm.

Table 11 shows the simulation results of experiment 1 to 6, with experiment 1 and 2 included to show differences of experiment 3 to 6 compared to the base experiments results. Figure 22 shows these results visually.

*Table 11: Trigger test phase experiments results.*

| Exp | New customers late [#] | Recurring customers late [#] | Violation time [min] | Total distance [km] | Switches made [#] | Transfers made [#] | CPU time for rescheduling calculations [s] |
|---|---|---|---|---|---|---|---|
| **1** | 214 | 4865 | 128665 | 354072 | 566 | 122 | 213 |
| **2** | 76 | 3761 | 119225 | 354466 | 3281 | 88 | 8499 |
| **3** | 81 | 3832 | 126633 | 355850 | 4165 | 81 | 9619 |
| **4** | 210 | 4721 | 127379 | 355540 | 1238 | 88 | 451 |
| **5** | 210 | 4729 | 128451 | 355269 | 1132 | 97 | 417 |
| **6** | 142 | 4218 | 115538 | 352723 | 1215 | 55 | 6446 |

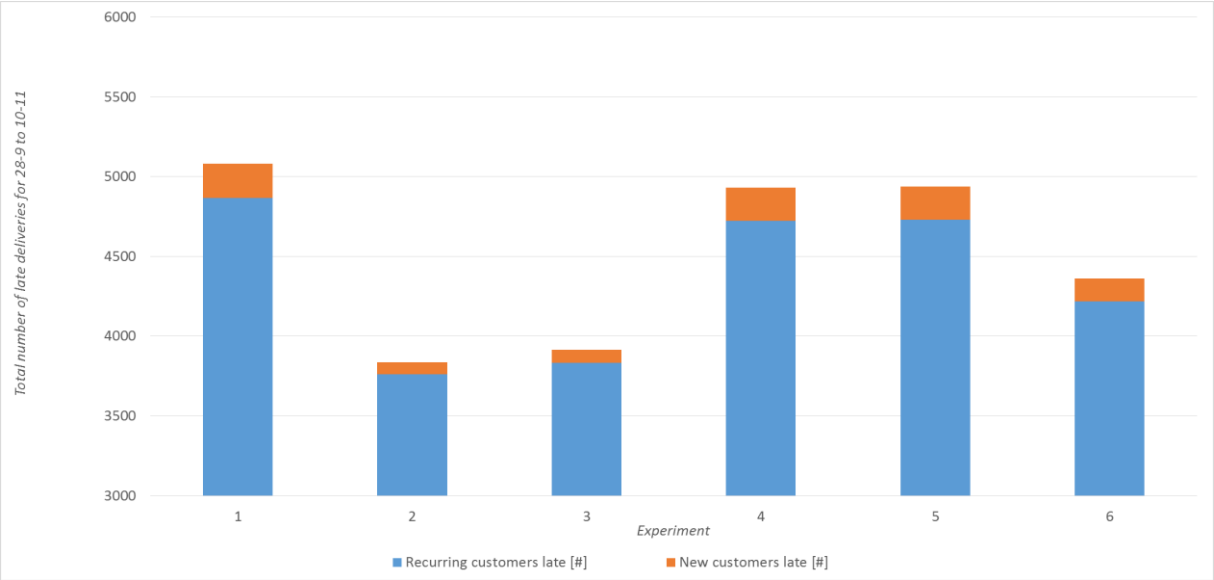


*Figure 22: late deliveries of experiment 1 to 6, with visual distinction between new and recurring customers.*

Our aim here is to test the effectiveness of the implemented triggers in the simulation. We see that varying the benchmark triggers and TSR triggers (experiment 1 vs 4) for the benchmark heuristic results in little difference in rescheduling results. The use of benchmark triggers in combination with the TSR method ensures an increase in late deliveries compared to the use of TSR triggers (experiment 6 vs 2). This indicates that the TSR triggers are best to use when for the TSR method.

The robustness trigger does not seem to significantly affect the rescheduling results (experiment 4 vs 5 and 2 vs 3) for both methods. Possible explanation is that delays are often linked to expected lateness, making the triggers similar in the sense that a delay will often ensure a trigger. Still, many more routes are changed when we use the TSR triggers. It therefore seems that the TSR triggers are more nervous, meaning they initiate rescheduling calculations more often. Most problems are triggered by the benchmark triggers as well, ensuring similar results in the end.

The number of intra-route switches increases a lot when using TSR triggers and the TSR method. More triggers occur and more than one rescheduling operation is allowed for each vehicle. We see that a lot less transfers are made in the TSR method, indicating that when an objective function that translates all parts of the delivery penalties and bonuses into time, transfers are generally less efficient than smart switching of customers. Although the increase in additional distance is significant and consistent for the TSR method, 2000 additional kilometers for 24 shifts to reduce this many late deliveries is reasonable. The CPU time increases a lot when using the TSR method.

## 6.3 OBJECTIVE FUNCTION

This section presents the results of experiments 7 to 21 to find the effects of different objective function configurations. Table 12 and Figure 23 show the simulation results of experiment 7 to 21. Recall that the following parameters are varied in these experiments:

- Experiment 2, 10, 14 and 18 use a linear penalty function for expected time window violations;
- Experiment 7, 11, 15 and 19 use a quadratic penalty function;
- Experiment 8, 12, 16 and 20 use hard time window penalties with 5 minutes 'grace';
- Experiment 9, 13, 17 and 21 use hard time window penalties with 15 minutes 'grace';
- Experiment 2, 7 to 9 and 14 to 17 use no robustness bonus while experiment 10 to 13 and 18 to 21 do apply a robustness bonus according to the robustness function as defined;
- Experiment 2 and 7 to 13 make a distinction between time window violations for new and recurring customers in the objective function while experiment 14 to 21 use the same penalties for both types of customers.

*Table 12: Objective function phase experiments results.*

| Exp | New customers late [#] | Recurring customers late [#] | Violation time [min] | Total distance [km] | Switches made [#] | Transfers made [#] | CPU time for rescheduling calculations [s] |
|---|---|---|---|---|---|---|---|
| **2** | 76 | 3761 | 119225 | 354466 | 3281 | 88 | 8499 |
| **7** | 71 | 3839 | 119302 | 354375 | 3200 | 97 | 8637 |
| **8** | 85 | 3909 | 127562 | 356530 | 3291 | 82 | 8555 |
| **9** | 74 | 3837 | 126879 | 354770 | 3144 | 70 | 8175 |
| **10** | 73 | 3910 | 119169 | 356071 | 3284 | 85 | 8911 |
| **11** | 71 | 3943 | 119336 | 356096 | 3260 | 75 | 8988 |
| **12** | 85 | 3822 | 125522 | 356098 | 3330 | 83 | 9028 |
| **13** | 79 | 3900 | 123908 | 356097 | 3252 | 45 | 9059 |
| **14** | 83 | 3940 | 119618 | 356051 | 3252 | 76 | 8297 |
| **15** | 85 | 3968 | 119489 | 356095 | 3208 | 88 | 8377 |
| **16** | 86 | 3905 | 127722 | 356552 | 3274 | 84 | 8539 |
| **17** | 92 | 4033 | 126426 | 356400 | 3138 | 46 | 8570 |
| **18** | 83 | 3895 | 119042 | 355992 | 3247 | 84 | 8455 |
| **19** | 81 | 3918 | 119357 | 356151 | 3216 | 75 | 8544 |
| **20** | 84 | 3824 | 125253 | 356113 | 3326 | 79 | 8662 |
| **21** | 90 | 3931 | 124328 | 356119 | 3226 | 42 | 8684 |

We aim to find good settings for our objective function. We find that the use of 15 minutes 'grace' and hard time windows in our time window violation penalty function results in the most customers receiving late deliveries and a large total time window violation time. The use of 5 minutes 'grace' and hard time windows decreases the number of time window violations, but it increases the total time window violation time compared to experiments where a type of soft time windows is used. We find that the linear penalty function experiments show more late deliveries for new customers and less for the recurring customers, when compared to the quadratic penalty function experiments. The linear penalty function can be considered to perform best of all penalty functions, since the total time window violation time is the lowest, while the number of late deliveries is low compared to the other types of penalty functions.

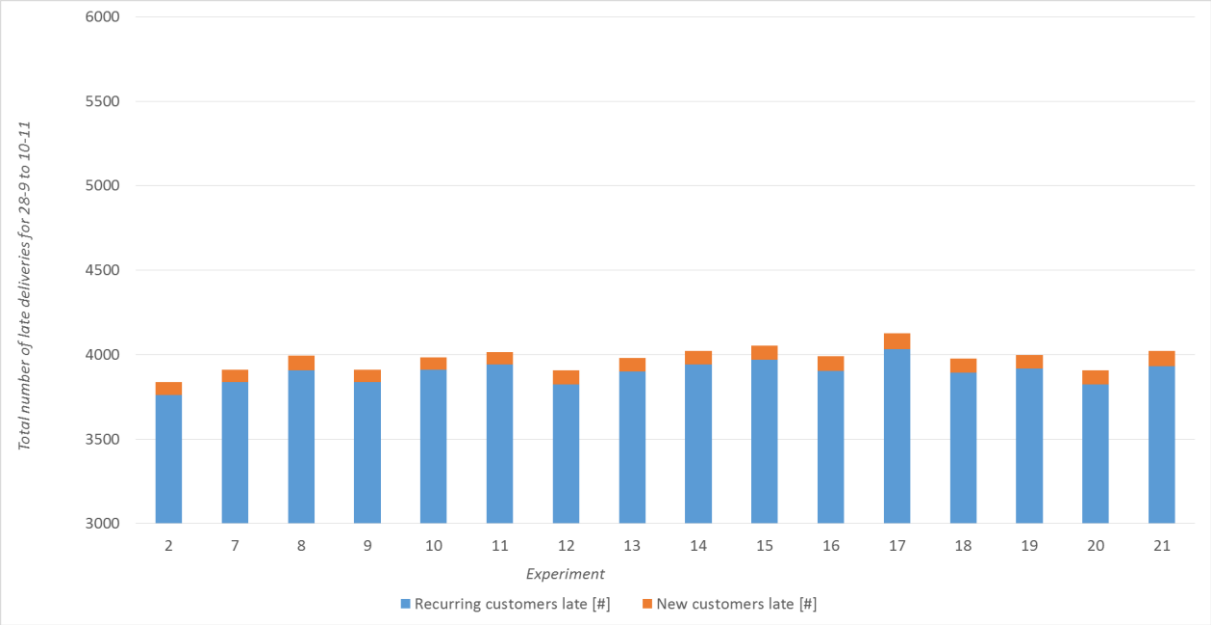


*Figure 23: late deliveries of experiment 7 to 21, with visual distinction between new and recurring customers.*

The use of a robustness bonus results in a decrease in the number of late deliveries and total time window violation time. Although differences are small, we find a consistent improvement for all experiments with the robustness function applied versus the experiment with all the same settings except that the robustness function is not applied. We recommend the use of our exponential robustness function in en-route rescheduling.

We experiment with the distinction between new and recurring customers in our penalty function. We see that the number of new customer late deliveries increases when we do not use the distinction between customer types in the penalty function. The number of late deliveries for recurring customers decreases slightly when we do not use customer distinction. The total time window violation time remains approximately the same when comparing an experiment with customer distinction to the same experiment without customer distinction. We recommend the use of new customer distinction. The customer distinction decreases the late deliveries for new customers (with higher priority), while no big sacrifice is necessary for the late delivery of the recurring customers. Note that this is based on a lot fewer new customers than recurring customers, so customer distinction might give different results in other cases.

For all variations of this phase we find similar numbers of transfers made, distance travelled and CPU time needed for the rescheduling calculations. Only the 15 minutes 'grace' hard time window penalty function shows a different number of transfers made, which can be explained by the fact that this penalty function accepts more violation, so more problems can be solved by switching only. Therefore, it is interesting to look at only using intra-route switching for solving en-route rescheduling problems in our managerial insights.

## 6.4 Operator selection

This section presents the results of experiments 22 to 26 to find the effects of different objective function configurations. Experiment 22 uses only or-opt moves (both intra-route and inter-route) and experiment 23 uses only exchange-moves. Experiment 24 to 26 use 50% exchange- and 50% or-opt-moves intra-route, and combinations of exchange-, or-opt-, and 2*-moves inter-route. Table 13 and Figure 24 show the simulation results of experiment 22 to 26.

*Table 13: Operator selection phase experiments results.*

| Exp | New customers late [#] | Recurring customers late [#] | Violation time [min] | Total distance [km] | Switches made [#] | Transfers made [#] | CPU time for rescheduling calculations [s] |
|---|---|---|---|---|---|---|---|
| **22** | 76 | 3744 | 120767 | 354964 | 3273 | 85 | 11550 |
| **23** | 74 | 3907 | 119084 | 356208 | 3290 | 10 | 3955 |
| **24** | 76 | 4004 | 122061 | 365113 | 3300 | 34 | 5962 |
| **25** | 78 | 3770 | 116871 | 356218 | 3344 | 28 | 7222 |
| **26** | 76 | 3756 | 121143 | 354430 | 3281 | 88 | 8647 |

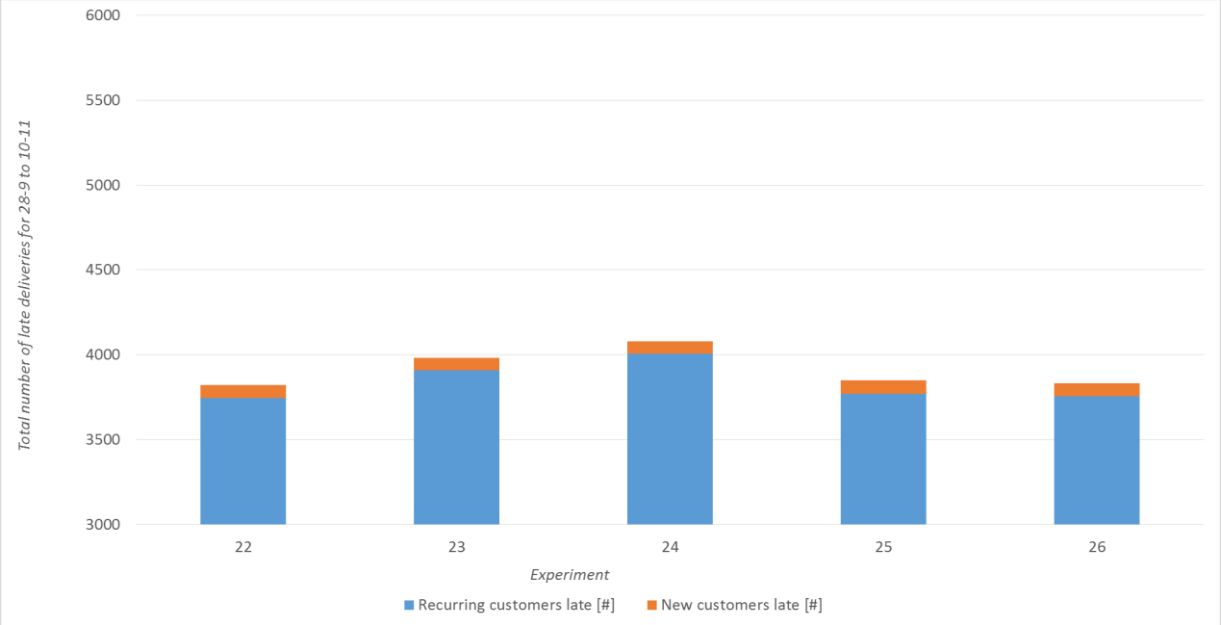


*Figure 24: late deliveries of experiment 22 to 26, with visual distinction between new and recurring customers.*

We look for the best operator settings for our TSR method here. From experiments 22 and 23 we see that the use of the Or-opt operator is superior to the use of the simple exchange-operator on numbers of late deliveries and total violation time. This requires a lot more CPU time, but we generally find better solutions. The exchange-operator does not seem to work well in the context of en-route rescheduling with time windows. The 2-opt*-operator does not seem to significantly affect the quality of the final solutions.

## 6.5 MANAGERIAL INSIGHTS

In this section, we aim find managerial insights on on-time departure, backup vehicle locations, combination of settings from the other phases, knowledge of future travel times, initial planning quality and phased TSR method implementation. We listed questions for further experimentation with our simulation model. For each of these questions, we describe how we test it with our simulation model. We show the experiment number for each of the additional experiments as well.

27. *What is the influence of on time departure for all vehicles in the current situation?* We test this by running an experiment, with the settings of our benchmark heuristic base experiment, without departure delays.

28. *What is the influence of adding additional slack time for the vehicles?* We test this by running an experiment, with the settings of our benchmark heuristic base experiment, where the vehicles leave 10 minutes before the planned departure.
29. *What are the effects of different, more logical, locations for backup vehicles?* We test this by placing backup vehicles in the centers of the three cities with the most late deliveries of the case study (see Appendix A): Amsterdam, Rotterdam and Den Haag.
30. *What are the potential benefits of our en-route rescheduling situation when using the best settings we find in each of the phases?* We run an experiment with the best experimental settings found in the previous phases: TSR triggers (without robustness trigger), a linear penalty function for time window violations, an exponential bonus function for more robust solutions, customer distinction between recurring and new customers, and the use of only or-opt-operators.
31. *What is the effect of the TSR method when we have no knowledge of future travel times at the moment of rescheduling?* We run an experiment with the settings of experiment 30 on a case where we know nothing about the future travel times at the moment of rescheduling.
32. *What is the effect the TSR method when we have complete knowledge of future travel times at the moment of rescheduling?* We run an experiment with the settings of experiment 30 on a case where we know all future travel times at the moment of rescheduling.
33. *What is the effect of the initial planning quality on the benchmark rescheduling heuristic results?* To test the effects of the intial planning we reschedule all routes with the intra-route part of the TSR method before departure. From this initial planning we assume we have closer to optimal routes with respect to the input we use in our simulation model. We use the benchmark rescheduling heuristic to find en-route rescheduling solutions in this experiment.
34. *What is the effect of the initial planning quality on the TSR method results?* Here, we reschedule all routes with our the intra-route part of our TSR method before departure as well. We use the TSR method to find en-route rescheduling solutions in this experiment with the settings of experiment 30 for this experiment. We compare the results of experiment 33 and 34 to find the benefits of the TSR method given that the quality of the intial plan is of less influence.
35. *What is the effect when we use only the en-route switching of customers part of the TSR method?* We run an experiment with the TSR method using the settings of experiment 30, where we only allow switch moves for the en-route rescheduling. When a breakdown occurs, a transfer of goods is still used, but no transfers are considered in the rest of the en-route rescheduling calculations.

Table 14 and Figure 25 (note that we use a different y-axis scale in this figure compared to previous figures) present the results of experiments 1, and 27 to 29. We run these experiments on the base experiment settings of the benchmark heuristic to study the influence of on time departure and backup-vehicle locations compared to the current situation (experiment 1). We find that on time departure (experiment 27) has a huge effect on the on time delivery and number of rescheduling operations needed. Of course, all vehicles departing on time is unrealistic and difficult to manage, but the impact of improvements on departure time level are large. We also see this impact in experiment 28, where we allow the vehicles to leave before the planned time. The impact of different backup vehicle locations is barely visible, as shown in experiment 29.

*Table 14: Managerial insights phase experiments results based on benchmark heuristic base experiment.*

| Exp | New customers late [#] | Recurring customers late [#] | Violation time [min] | Total distance [km] | Switches made [#] | Transfers made [#] | CPU time for rescheduling calculations [s] |
|---|---|---|---|---|---|---|---|
| **1** | 214 | 4865 | 128665 | 354072 | 566 | 122 | 213 |
| **27** | 126 | 3045 | 49702 | 352775 | 266 | 105 | 81 |
| **28** | 80 | 1769 | 27628 | 352144 | 121 | 83 | 43 |
| **29** | 214 | 4851 | 125527 | 353862 | 555 | 132 | 216 |

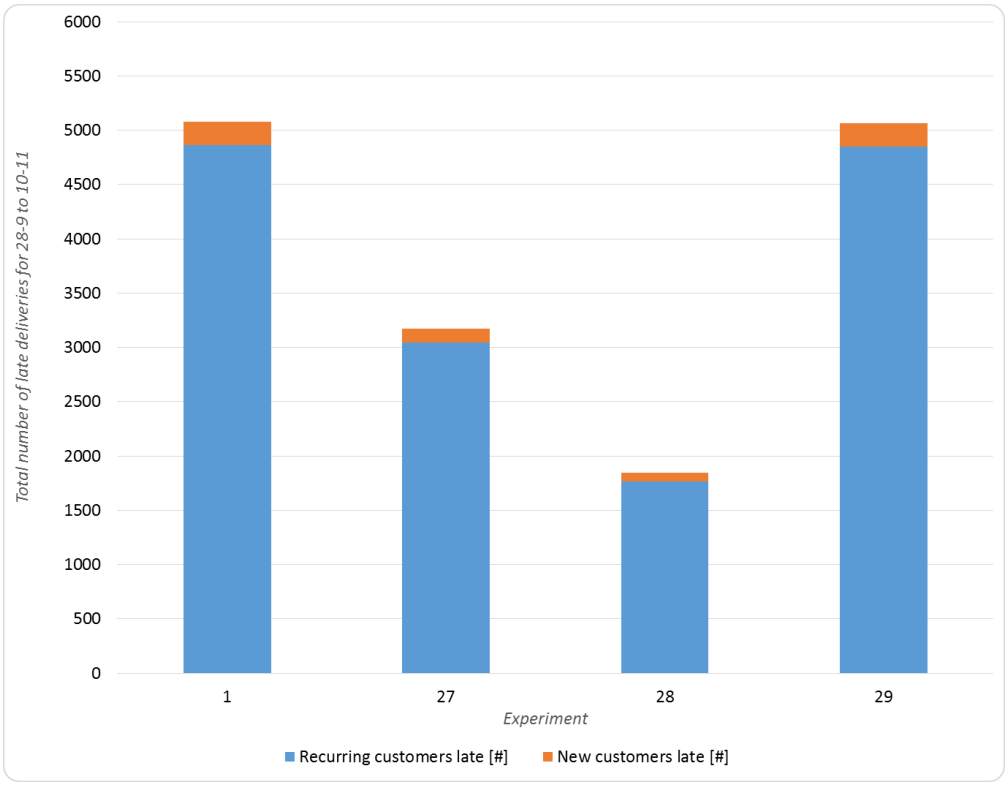


*Figure 25: number of late deliveries for experiment 1, and 27 to 29 to show the effects of on time departure (experiment 27), extra slack (experiment 28) and different backup vehicle locations (experiment 29).*

In Section 6.2 to 6.4, we found that TSR triggers without robustness trigger, a linear penalty function, use of a robustness function, new customer distinction and the use of only or-opt moves are the best settings for the case study problem. We therefore run these settings as experiment 30 to find the potential savings if we use the settings we found in the previous phases. We find that the combination of thes settings from each of the phases shows the best results so far for our TSR method. We are able to decrease the number of late deliveries with 63% for new customers and 24% for recurring customers. We display the decrease in the total number of customer (new and recurring) in Figure 26. The total time window violation time is reduced with 9% in this experiment. Only 0.3% extra distance needs to be travelled for this, but more rescheduling calculations are necessary. The total number of changes is about 5 times higher, but these changes are mostly intra-route. Furthermore, we determine the changes automatically, which ensures that no manual determination of en-route rescheduling solutions is needed for a control room.

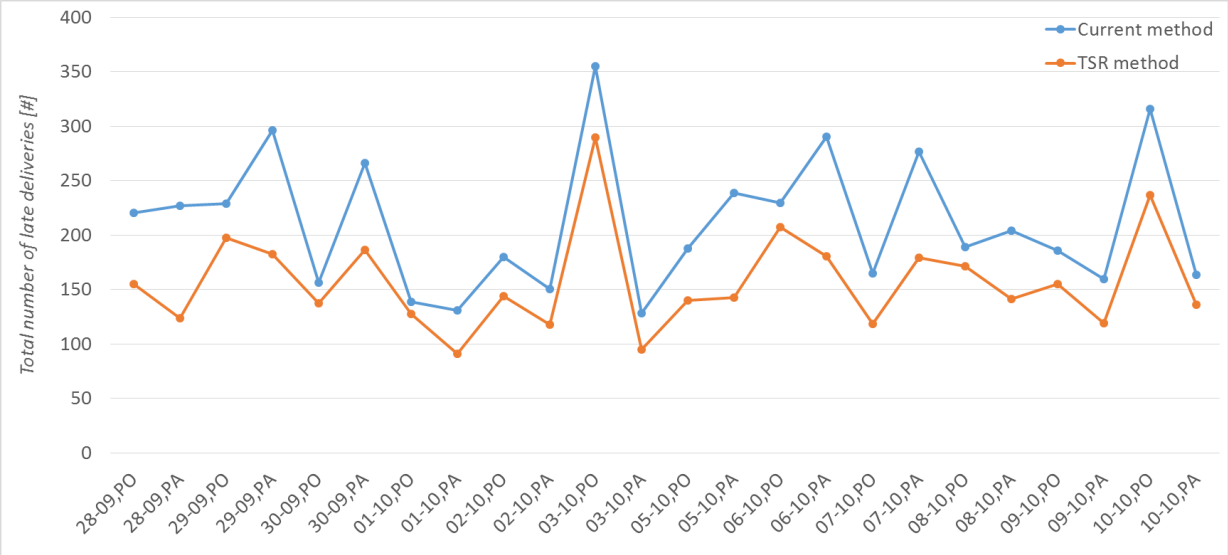


*Figure 26: total number of late deliveries (sum of new and recurring customers) for every day in the simulation comparing the current situation and the best settings for the TSR method.*

We show the results of experiment 30 to 35 in Table 15 and Figure 27, where bothFigure 27 also display the basic benchmark experiment to show the improvements compared to the current situation. We find that the effect of our rescheduling method when we have no knowledge of future travel times (experiment 31) at the moment of rescheduling is minor when compared to the current situation (experiment 1). Although we find good improvements for the number of new customers that receives a late delivery, the other variables show only a minor improvement when we know knothing about the future travel times at the moment of rescheduling. When we would be able to accurately predict all travel times at the moment of rescheduling, our TSR method shows much more improvement. The experiment with complete knowledge of future travel times (experiment 32) shows that we even further reduce the number of late deliveries when we can accurately predict all future travel times.

When researching the effects of initial planning quality, we test pre-departure intra-route TSR method optimization of routes in combination with the benchmark rescheduling heuristic (experiment 33) and en-route TSR method (experiment 34). We find that the initial planning quality influences the number of late deliveries for both methods. When comparing experiment 1 (which models the current situation) and experiment 33 we see a large decrease in numbers of late deliveries for recurring customers when the initial planning quality effect is removed. We see virtually no effects when comparing experiment 30 and 34 however. This means that for the TSR method, pre-departure optimization is less relevant. This can probably be explained by the fact that the triggers would go off in this method soon enough when the vehicles depart from the depot in this method. The use of the TSR method for en-route rescheduling is still effective when pre-departure rescheduling is used to remove initial planning quality effects. The TSR method (experiment 34) results in less late deliveries than the benchmark rescheduling heuristic (experiment 33). However, we do find that the difference in late deliveries is smaller than when the planning instances are not rescheduled pre-departure.

Finally, we test the effect of only implementing the intra-route switching part of the TSR method in experiment 35 to see the results of when a phased implementation of the TSR method is used in the DCT. Furthermore, as stated in Section 6.3, few transfers are made in the TSR method anyway. We see that the number of late deliveries is close to the total found in experiment 30, with only a slight increase in late deliveries for recurring customers. The number of transfers that are used is exactly equal to the number of breakdowns (see also Appendix D), so transfers are used only when a vehicle cannot continue. Based on these results we recommend a phased implementation.

*Table 15: Managerial insights phase experiments results based on best settings for TSR algorithm.*

| Exp | New customers late [#] | Recurring customers late [#] | Violation time [min] | Total distance [km] | Switches made [#] | Transfers made [#] | CPU time for rescheduling calculations [s] |
|---|---|---|---|---|---|---|---|
| **1** | 214 | 4865 | 128665 | 354072 | 566 | 122 | 213 |
| **30** | 80 | 3692 | 118013 | 355117 | 3258 | 86 | 11499 |
| **31** | 98 | 4462 | 124551 | 355500 | 3118 | 84 | 11376 |
| **32** | 75 | 3570 | 114403 | 354960 | 3225 | 81 | 12103 |
| **33** | 238 | 3886 | 122062 | 350776 | 399 | 128 | 165 |
| **34** | 80 | 3672 | 112051 | 357060 | 2774 | 90 | 10528 |
| **35** | 75 | 3740 | 114252 | 356937 | 3330 | 4 | 7971 |

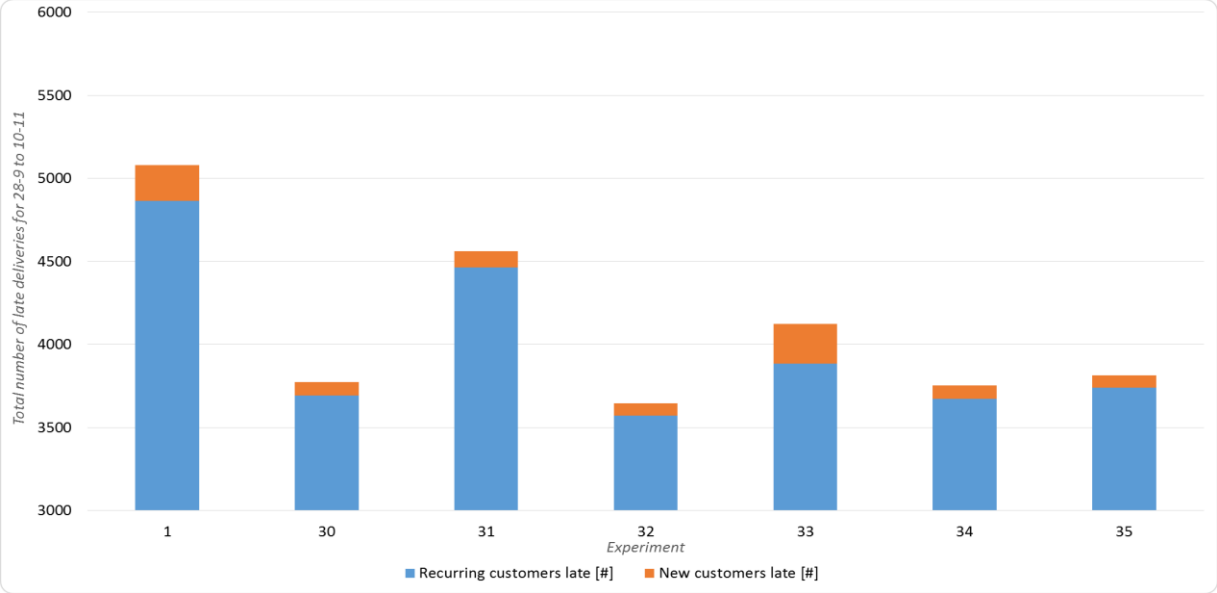


*Figure 27: number of late deliveries for experiment 1, and 30 to 35 to show the effects of the best TSR method settings (30), knowledge of future travel times (31 and 32), intial planning quality (33 and 34), and a phased implementation (35).*

## 6.6 RESULTS CONCLUSION

Our TSR method is able to reduce the number of late deliveries with 63% for new customers and 24% for recurring customers in a realistic simulation. No more manual work by control room teams is involved because we automatically trigger and calculate rescheduling solutions that can be implemented in a decision support system. We find that the TSR method works best with TSR triggers (without robustness trigger), a linear penalty function for time window violations, an exponential bonus function for more robust solutions, customer distinction between recurring and new customers, and the use of only or-opt-operators in our method. We show that we are able to reduce the late deliveries for all shifts in the test set based on our simulation model.

We that on time departure can reduce the number and severity of late deliveries. We show that the effect of knowledge on future travel times is significant. Large differences are found when experimenting between no and complete knowledge of future travel times. Initial planning quality is of large influence on the number of late deliveries, but for en-route rescheduling we find that the TSR method still outperforms the benchmark rescheduling heuristic. We show that we are able to solve most rescheduling problems by only intra-route switching, so we recommend a phased implementation. Transfers should be used when this is really necessary, which is often not the case.

# 7 CONCLUSIONS & DISCUSSION

This chapter aims to answer the research question and discuss our findings. First, we answer the research questions and draw conclusions with respect to our research goal in Section 7.1. Next, we discuss the method and test simulation of our study in Section 7.2.

## 7.1 CONCLUSIONS

In this section, we describe our conclusions with respect to our research goal of developing a methodology for en-route rescheduling of home deliveries. To achieve the research goal, we defined multiple research questions in Chapter 1. We answer each of these questions individually here, to eventually come to a general conclusion on our methodology for en-route rescheduling of home deliveries.

*1) What is the current state of research on en-route rescheduling related subjects?*

We find that literature on vehicle rerouting for road transport is scarce. The studies on the subject that do exist mainly consider vehicle breakdowns and changing customer wishes, which differs from our focus in the home delivery operation case study. Triggers for rescheduling and the transfer of goods from one vehicle to another, as the case is in this research project, seems to be new in the use case of en-route rescheduling of home deliveries. In existing literature on the subject, variants of Tabu Search are often applied because this metaheuristic is able to find good solutions fast. We contribute to the current state of en-route rescheduling research by researching en-route rescheduling in home delivery operations with violation triggers, transfers and time windows.

*2) What is the current rescheduling process in the home delivery case study?*

The current rescheduling process is designed in such a way that a control room team supervises a delivery operation. They think of solutions when a driver calls that he is running more than 20 minutes late. Much added value is already found when their decision making process can be supported by automated rescheduling calculations, because the current rescheduling process is labor-intensive. When a control room needs to find a rescheduling solution they check if they can solve the problem by moving possibly late stops forward in time. When this does not solve the existing problem, he checks for empty vehicles near the disrupted vehicle to see if another vehicle is available to help.

*3) How can we ensure robust en-route rescheduling of home deliveries?*

Based on the current process and the literature review, we find that a combination of triggers, search space selection and Tabu Search for the best rescheduling solution is interesting for en-route rescheduling solutions. We use triggers to determine when we think it is necessary to reschedule the current course of events. Possible triggers are: delays, expected lateness, possible lateness (which includes a robustness trigger), vehicle defects, vehicle breakdowns, and accidents or events on the planned path of the vehicle. For testing purposes, we test only the eta, vehicle defect and vehicle breakdown triggers in this study.

When we trigger for rescheduling, we select vehicles we want to evaluate for helper-actions. We determine the possible helper vehicles using an equation based on helper distance and the number of deliveries to be made by the helper. We evaluate all helper actions by backup and en-route vehicles simultaneously with intra-route switching of customers. We determine the solution quality of the initial solution, all possible helper actions and the intra-route switching using an objective function. We select the solution with the lowest cost to continue the routes of vehicles. The objective function

consists of total time needed for the vehicles, time window violation penalties, and a possible robustness bonus function. We translate all variables into time to get a single result from our function.

*4) How can we test rescheduling performance of the developed algorithm?*

To test the TSR method performance we made a simulation model. The simulation model aims to find good TSR method settings and to show potential savings of our TSR method. We run the simulation model on two weeks of data, meaning that we run 24 morning and afternoon shifts for all tested TSR method settings. We model all activities of the delivery vehicles, and we verify and validate the simulation model on realizion data from the DCT. We verify that the model is implemented correctly. We validate the model and find that, although more late deliveries follow from our model, the general pattern is similar to the real situation. The total time fits the real situation well and the numbers of late deliveries are in the same order of magnitude as in the real situation. The simulation model enables us to show the results of our TSR method compared to the current situation.

*5) What performance can we expect from the methods we develop?*

We find that the TSR method works best with TSR triggers (without robustness trigger), a linear penalty function for time window violations, an exponential bonus function for more robust solutions, customer distinction between recurring and new customers, and the use of only or-opt-operators in the Tabu Search processes. Using these settings, we find from our simulation that we can potentially decrease the number of late deliveries with 63% for new customers and 24% for recurring customers. The total time window violation time is reduced with 9%. Only little additional distance is travelled to do this, but a lot more rescheduling calculations are necessary. The total number of changes made en-route is around 5 times higher.

We also see that on time departure can reduce the number and severity of late deliveries. When possible, the retailer should even aim to depart earlier to have some slack in the route planning. We see a large decrease in lateness when we do this. We show that the effect of knowledge on future travel times is large. Initial planning quality is of big influence on the number of late deliveries, but for en-route rescheduling we find that the TSR method still outperforms the benchmark rescheduling heuristic. We show that we are able to solve most rescheduling problems by only intra-route switching, so we recommend a phased implementation. From our simulation study we find that transfers are only rarely better than intra-route changes in a home delivery process.

In general, we conclude that we successfully developed a method for en-route rescheduling of home deliveries. Our TSR method is able to greatly reduce the number of late deliveries by calculating the best intra-route and inter-route changes. No more manual work by control room teams is involved because we automatically trigger and calculate rescheduling solutions that can be implemented in a decision support system.

## 7.2 Discussion

In this section we discuss the method and the simulation model. After this, we set our directions for future research on the subject of en-route rescheduling. We end this chapter with recommendations for practical implementation of the solutions.

### 7.2.1 Discussion

We discuss the TSR method and the simulation setup here. We give an overview of possible triggers, but we were not able to test all of them. The cause of a trigger can be different in some cases, meaning that it should be treated differently for the rescheduling input. We left out some real-world constraints

to reduce computation time. Still, some real-world constraints for the rescheduling problem, such as the capacity of vehicles and driving time regulations for the drivers, might influence solutions and solution quality as well in reality. We tested a few operators in the en-route rescheduling and found some interesting results for this subject. Still, more operators exist in reality that might be interesting to include in future testing as well. Because of the pre-selection we make based on literature, we think we incorporated the most relevant operators for testing in our research.

In our research we used an objective function where we translate all variables into time. This means that we use travel time as our main variable, but it also means that we translate all penalties and bonuses into time variables. This can ensure that we can compensate expected lateness (which is expressed in a penalty) with either a shorter total route time or a bonus that we apply for finding a more robust solution for other customers. We therefore tuned the bonus functions to give a much lower bonus to a robust solution than the penalty that we apply for expected time window violations, but it is possible that we compensate lateness with other variables in the objective function.

An important discussion point of this study is the degree of realism of our simulation setup. We conclude that the algorithm we designed can potentially ensure large savings, but this can also be caused by the way that we set up the simulation. We believe simulation is the best choice for this preliminary study, but tests of actual implementation can always give different results. It was not possible to translate all actions of the control room team into an algorithm, because it is difficult to model such human behavior. Furthermore, we did not have full information on which rescheduling actions are performed at the moment, meaning that we could not validate the rescheduling actions white-boxed. Small differences show between the simulation and real-world situation, but a general pattern is modelled well in our simulation. We can say that, although the savings might differ in real-world implementation, we believe to show a realistic insight in what the savings from the use of our Tabu Search Rescheduling (TSR) method might be.

We apply our test simulation on real-life planning instances of our case study retailer. These planning instances are based on different data sources than what we can access in this study. Another issue of using the real-world planning instances is that the savings we find depend on the initial planning quality. When many illogical customer sequences exist in the initial planning instances, more improvements can be found when en-route rescheduling it triggered. Based on our initial planning quality experiments we can still find a reduction in the number of late deliveries, meaning that the results we find are not only caused by bad planning instances. However, the effects of our TSR method are smaller in when using pre-departure rescheduling.

Another point of discussion is that we use two weeks of data for our simulation runs. We select these specific weeks based on their completeness of the data and registrations. Furthermore, the weeks we select are not holidays or special weeks in another way, meaning that they will be representative for the quality of our method when it is applied on a different time period. The service of the retailer is quickly growing however, so the future situation might be that many more deliveries are made. The method, and the product we aim to use for application of our method, are general in the sense that it is possible to tune the algorithm for other use cases, for example a retailer that does not wish to apply different customer priorities in its delivery service.

Final point of discussion is that we use an approximation of the travel times to reduce the calculation times within our experiments. Because we could not model the entire road network and possible paths due to limited time and resources, we use travel times based on Euclidian distance classes. A random deviation is based on realized deviation from the realization data. Although we apply the same delays to all the paths towards the following five stops from the current location, so our solution prediction

is not influenced by the randomness, the assumption that we can predict the travel times to the next five stops precisely and that the rest of the travel times is completely unpredictable is somewhat unrealistic. The realism of the test simulation might improve by using better travel time information. No information of the time of the day and the possible traffic situation is included in our approximation. Because of our assumptions on travel times, we also assumed that travel delays are not correlated, but the question might be if the delay is caused by traffic or other activities (e.g., finding a parking space and administrative actions by a driver). Our travel time assumptions cause delays to be a little more random than the real-world situation. We do believe that the results we find are a good indication of the savings that apply when using real-world travel times and travel time predictions.

### 7.2.2 Directions for future research

Because only little research exists on the subject of en-route rescheduling, a lot of interesting directions for future research can be set out based on our research. We introduce many new aspects to this subject, but a lot of them might require more research in future projects.

We give an overview of possible triggers and test some of the them. It would be interesting to test other trigger types in a simulation context, such as the accident or event triggers, where geographic areas are affected by delays. Another interesting topic of research can be to trigger rescheduling calculations when vehicles are way too early, meaning they can alter their route to reduce the total time needed in planning instances where (too) much slack is used.

Besides triggers, a form of continuous calculation of possible improvements can be interesting for future research as well. In this type of methods, possible intra-route and inter-route improvements can be calculated continuously, and when more than a threshold improvement is found, this can be proposed in the same decision support way as described for our research.

In our research, we select a certain type of search space selection to define possible vehicles for inter-route changes when triggered for en-route rescheduling. We did not have the time and resources to check if this is the best way to find relevant helper-vehicles. More research on good search space selection could help make the en-route rescheduling, and even the more general DVRP and VRPTW problems be solved faster and more efficiently.

We choose to use Tabu Search as our local search algorithm in the rescheduling calculations based on the literature that is already available on the subject, but other metaheuristics might be interesting for future research as well. The characteristic of Tabu Search, that it finds good solutions fast, makes it suitable for en-route rescheduling, but when using a different metaheuristic good solution might be found as well. Especially when combined with continuous calculation of improvements, more time is available for rescheduling calculations, making it possible that other metaheuristics are interesting for en-route rescheduling.

We apply some objective functions in this research that can be interesting for the more general VRPTW problem as well. Different soft time window violation functions are often tested in literature, but the application of a robustness function can be interesting when much uncertainty in vehicle routing instances exists. Furthermore, the customer priority we use for new customers can be interesting to research further for different priority-classes as well. We show that we can effectively use customer priority without harming the general solution quality too much in our en-route rescheduling simulation. This makes the customer priority part an interesting subject to research in the future, possibly in combination with a different robustness function for customer priority.

The use of simulation to test the robustness is necessary to see the effects of different objective functions in a delivery process with uncertainty. This research direction is interesting for future research, since the late deliveries depend largely on the uncertainty in the delivery process. A larger and more time-consuming simulation of realistic road network and realistic traffic data is interesting in this context as well.

Another interesting direction for future research is real-world testing and implementation. When the method is implemented and used as we will describe in the next section, it is interesting to evaluate the number and severity of late deliveries by the retailer. The results of this are relevant for future research and implementation of rescheduling methods.

### 7.2.3 Recommendations for implementation in the DCT

This research project is initiated by Simacan to find the added value of a Decision Support System (DSS) in their Delivery Control Tower (DCT). The case study retailer is interested in the outcomes of our research as well, since it might potentially decrease their number of late deliveries and because it can save their control room team much time and effort. Because of this practical added value of the project, we describe the recommendations for implementation in the DCT here.

We showed in this study that the TSR method as a whole has a lot of potential to decrease the number and severity of late deliveries in a home delivery operation. Because the implementation of the entire solution is time-consuming and potentially difficult, we propose a light-weight version of the rescheduling method.

Arrival times are already predicted for each of the stops on a route in the DCT, so the TSR triggers can be easily implemented as described for the simulation setup. Still, it should be possible to manually request a rescheduling calculation, since defects and breakdowns will still be communicated by phone. The difficulty of actual implementation will exist mostly in the calculation of transfers. This is the most difficult part of our TSR method. As shown in the results of this study, only a few transfers are made as a result of this difficult methodology. The added value of all these extra calculations is relevant, but the time and difficulty of implementation might mean that the benefits of implementing this part of the solution do not outweigh the cost of implementation.

We recommend to only implement the TSR method switching solution with the best settings we find for the TSR method first. This way, many potentially late deliveries are automatically triggered and the switching algorithm ensures that good solutions are proposed automatically by the DSS implemented in the DCT. We prove to find the largest reduction in late deliveries by only implementing our switching solution, while the transfers only add little additional decrease when compared with the intra-route switching. Implementation of the inter-route transfers is also lucrative, but implementation is a lot more difficult and the need for a transfer occurs less frequent. In the more exceptional cases, transfers can still be determined manually by the control room.

With the implementation of our TSR method, a control room has to devote less time and effort to find rescheduling solutions, and they can just either accept or reject the solution. This ensures that they are able to focus on other urgent tasks in the delivery process. A large part of the possibly late deliveries can be automatically prevented using the TSR method.

# REFERENCES


Bertsimas, D. J., & Simchi-Levi, D. (1996). A new generation of vehicle routing research: robust algorithms, addressing uncertainty. *Operations Research, 44*(2), 286-304.

Bouros, P., Sacharidis, D., Dalamagas, T., & Sellis, T. (2011). Dynamic Pickup and Delivery with Transfers. In D. Pfoser, Y. Tao, K. Mouratidis, M. Nascimento, M. Mokbel, S. Shekhar, & Y. Huang (Eds.), *Advances in Spatial and Temporal Databases* (Vol. 6849, pp. 112-129): Springer Berlin Heidelberg.

Bräysy, O. (2002). Fast local searches for the vehicle routing problem with time windows. *INFOR, 40*(4), 319-330.

Bräysy, O., & Gendreau, M. (2005a). Vehicle routing problem with time windows, Part I: Route construction and local search algorithms. *Transportation science, 39*(1), 104-118.

Bräysy, O., & Gendreau, M. (2005b). Vehicle routing problem with time windows, Part II: Metaheuristics. *Transportation science, 39*(1), 119-139.

Calvo, R. W. (2000). A new heuristic for the traveling salesman problem with time windows. *Transportation science, 34*(1), 113-124.

Caseau, Y., & Laburthe, F. (1999). Heuristics for Large Constrained Vehicle Routing Problems. *Journal of Heuristics, 5*(3), 281-303. doi:10.1023/A:1009661600931

Clarke, G., & Wright, J. W. (1964). Scheduling of Vehicles from a Central Depot to a Number of Delivery Points. *Operations Research, 12*(4), 568-581. doi:10.2307/167703

Cordeau, J. F., Gendreau, M., Laporte, G., Potvin, J. Y., & Semet, F. (2002). A Guide to Vehicle Routing Heuristics. *The Journal of the Operational Research Society, 53*(5), 512-522. doi:10.2307/823019

Cordeau, J. F., Laporte, G., & Mercier, A. (2004). Improved Tabu Search Algorithm for the Handling of Route Duration Constraints in Vehicle Routing Problems with Time Windows. *The Journal of the Operational Research Society, 55*(5), 542-546. doi:10.2307/4101915

Cordeau, J. F., Laporte, G., Savelsbergh, M. W. P., & Vigo, D. (2006). Vehicle routing. *Transportation, handbooks in operations research and management science, 14*, 367-428.

Cordone, R., & Calvo, R. W. (2001). A Heuristic for the Vehicle Routing Problem with Time Windows. *Journal of Heuristics, 7*(2), 107-129. doi:10.1023/A:1009665907495

Cortés, C. E., Matamala, M., & Contardo, C. (2010). The pickup and delivery problem with transfers: Formulation and a branch-and-cut solution method. *European Journal of Operational Research, 200*(3), 711-724. doi:http://dx.doi.org/10.1016/j.ejor.2009.01.022

Dantzig, G. B., & Ramser, J. H. (1959). The Truck Dispatching Problem. *Management Science, 6*(1), 80-91. doi:10.2307/2627477

Dorigo, M., Caro, G. D., & Gambardella, L. M. (1999). Ant algorithms for discrete optimization. *Artificial life, 5*(2), 137-172.

El-Sherbeny, N. A. (2010). Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. *Journal of King Saud University - Science, 22*(3), 123-131. doi:http://dx.doi.org/10.1016/j.jksus.2010.03.002

Ferrucci, F., & Bock, S. (2014). Real-time control of express pickup and delivery processes in a dynamic environment. *Transportation Research Part B: Methodological, 63*, 1-14. doi:http://dx.doi.org/10.1016/j.trb.2014.02.001

Ferrucci, F., & Bock, S. (2015). A general approach for controlling vehicle en-route diversions in dynamic vehicle routing problems. *Transportation Research Part B: Methodological, 77*(0), 76-87. doi:http://dx.doi.org/10.1016/j.trb.2015.03.003

Ferrucci, F., Bock, S., & Gendreau, M. (2013). A pro-active real-time control approach for dynamic vehicle routing problems dealing with the delivery of urgent goods. *European Journal of Operational Research, 225*(1), 130-141. doi:http://dx.doi.org/10.1016/j.ejor.2012.09.016

Gambardella, L. M., Taillard, É., & Agazzi, G. (1999). *Macs-vrptw: A multiple colony system for vehicle routing problems with time windows.* Paper presented at the New ideas in optimization.

Gendreau, M. (Producer). (2002). Metaheuristics in vehicle routing. [Powerpoint slides]
Gendreau, M., Hertz, A., & Laporte, G. (1992). New insertion and postoptimization procedures for the traveling salesman problem. *Operations Research, 40*(6), 1086. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&db=bsh&AN=4495319&site=ehost-live
Gendreau, M., Hertz, A., & Laporte, G. (1994). A Tabu Search Heuristic for the Vehicle Routing Problem. *Management Science, 40*(10), 1276-1290. doi:doi:10.1287/mnsc.40.10.1276
Gendreau, M., & Potvin, J.-Y. (1998). *Dynamic vehicle routing and dispatching*: Springer.
Gillett, B. E., & Miller, L. R. (1974). A Heuristic Algorithm for the Vehicle-Dispatch Problem. *Operations Research, 22*(2), 340-349. doi:10.2307/169591
Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research, 13*(5), 533-549.
Holland, J. H. (1975). Adaptation in natural and artificial systems.
Huisman, D., Freling, R., & Wagelmans, A. P. M. (2004). A robust solution approach to the dynamic vehicle scheduling problem. *Transportation science, 38*(4), 447-458.
Ichoua, S., Gendreau, M., & Potvin, J.-Y. (2000). Diversion Issues in Real-Time Vehicle Dispatching. *Transportation science, 34*(4), 426-438. doi:doi:10.1287/trsc.34.4.426.12325
Jaillet, P., & Wagner, M. R. (2008). Generalized online routing: New competitive ratios, resource augmentation, and asymptotic analyses. *Operations Research, 56*(3), 745-757.
Kirkpatrick, S. (1984). Optimization by simulated annealing: Quantitative studies. *Journal of statistical physics, 34*(5-6), 975-986.
Klundert, J. v. d., & Wormer, L. (2010). ASAP: The After-Salesman Problem. *Manufacturing & Service Operations Management, 12*(4), 627-641. doi:doi:10.1287/msom.1100.0292
Kok, A. L., Meyer, C. M., Kopfer, H., & Schutten, J. M. J. (2010). A Dynamic Programming Heuristic for the Vehicle Routing Problem with Time Windows and European Community Social Legislation. *Transportation science, 44*(4), 442-454. doi:doi:10.1287/trsc.1100.0331
Law, A. (2007). Simulation modeling and analysis. *McGraw-Hill series in industrial engineering and management science Show all parts in this series*.
Li, J.-Q., Borenstein, D., & Mirchandani, P. B. (2007). A decision support system for the single-depot vehicle rescheduling problem. *Computers & Operations Research, 34*(4), 1008-1032.
Li, J.-Q., Mirchandani, P. B., & Borenstein, D. (2007). The vehicle rescheduling problem: Model and algorithms. *Networks, 50*(3), 211-229. doi:10.1002/net.20199
Li, J.-Q., Mirchandani, P. B., & Borenstein, D. (2009). Real-time vehicle rerouting problems with time windows. *European Journal of Operational Research, 194*(3), 711-727. doi:http://dx.doi.org/10.1016/j.ejor.2007.12.037
Lin, S. (1965). Computer solutions of the traveling salesman problem. *Bell System Technical Journal, The, 44*(10), 2245-2269.
Masson, R., Lehuédé, F., & Péton, O. (2013). An Adaptive Large Neighborhood Search for the Pickup and Delivery Problem with Transfers. *Transportation science, 47*(3), 344-355. doi:doi:10.1287/trsc.1120.0432
Masson, R., Lehuédé, F., & Péton, O. (2014). The Dial-A-Ride Problem with Transfers. *Computers & Operations Research, 41*(0), 12-23. doi:http://dx.doi.org/10.1016/j.cor.2013.07.020
Mirchandani, P., Li, J.-Q., & Hickman, M. (2010). A macroscopic model for integrating bus signal priority with vehicle rescheduling. *Public Transport, 2*(3), 159-172. doi:10.1007/s12469-010-0028-3
Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research, 24*(11), 1097-1100. doi:http://dx.doi.org/10.1016/S0305-0548(97)00031-2
Mu, Q., Fu, Z., Lysgaard, J., & Eglese, R. (2011). Disruption management of the vehicle routing problem with vehicle breakdown. *The Journal of the Operational Research Society, 62*(4), 742-749. doi:10.2307/41059132
Ohlmann, J. W., & Thomas, B. W. (2007). A compressed-annealing heuristic for the traveling salesman problem with time windows. *INFORMS Journal on Computing, 19*(1), 80-90.

Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A. L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research, 225*(1), 1-11. doi:http://dx.doi.org/10.1016/j.ejor.2012.08.015

Potvin, J.-Y., & Rousseau, J.-M. (1995). An exchange heuristic for routeing problems with time windows. *Journal of the Operational Research Society, 46*(12), 1433-1446.

Respen, J., Zufferey, N., & Potvin, J.-Y. (2014). *Online vehicle routing and scheduling with continuous vehicle tracking.* Paper presented at the ROADEF-15ème congrès annuel de la Société française de recherche opérationnelle et d'aide à la décision.

Robinson, S. (2004). *Simulation: the practice of model development and use*. Chichester: John Wiley & Sons.

Rochat, Y., & Taillard, É. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics, 1*(1), 147-167. doi:10.1007/BF02430370

Savelsbergh, M. W. P. (1992). The Vehicle Routing Problem with Time Windows: Minimizing Route Duration. *ORSA Journal on Computing, 4*(2), 146. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&db=bah&AN=4480089&site=ehost-live

Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., & Dueck, G. (2000). Record Breaking Optimization Results Using the Ruin and Recreate Principle. *Journal of Computational Physics, 159*(2), 139-171. doi:http://dx.doi.org/10.1006/jcph.1999.6413

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems *Principles and Practice of Constraint Programming—CP98* (pp. 417-431): Springer.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research, 35*(2), 254-265.

Spliet, R., Gabor, A. F., & Dekker, R. (2014). The vehicle rescheduling problem. *Computers & Operations Research, 43*(0), 129-136. doi:http://dx.doi.org/10.1016/j.cor.2013.09.009

Thangiah, S., Fergany, A., & Awan, S. (2007). Real-time split-delivery pickup and delivery time window problems with transfers. *Central European Journal of Operations Research, 15*(4), 329-349. doi:10.1007/s10100-007-0035-x

Treiber, M., & Kesting, A. (2013). Traffic flow dynamics. *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*.

Visentini, M., Borenstein, D., Li, J.-Q., & Mirchandani, P. (2014). Review of real-time vehicle schedule recovery methods in transportation services. *Journal of Scheduling, 17*(6), 541-567. doi:10.1007/s10951-013-0339-8

Wang, X., Ruan, J., & Shi, Y. (2012). A recovery model for combinational disruptions in logistics delivery: Considering the real-world participators. *International Journal of Production Economics, 140*(1), 508-520. doi:http://dx.doi.org/10.1016/j.ijpe.2012.07.001

# Appendices

*En-route rescheduling of home delivery routes*

*---------- Confidential ----------*