Robustness increase in plans, a case study for home-delivery services

Master Thesis

Industrial Engineering & Management

23 September 2022

AUTHOR

Ing. Michiel Mol

EXAMINATION COMMITTEE

Dr.ir. E.A. Lalla-Ruiz

Dr.ir. M.R.K. Mes

UNIVERSITY OF TWENTE

EXTERNAL SUPERVISOR

Ir. Jeroen Soesbergen

SIMACAN

Preface

This master thesis represents the end of my educational career and time at the University of Twente. Despite the harsh Covid times I encountered during the whole master I can look back at a pleasant time. My career consists of a bachelor Industrial Engineering & Management at Windesheim University of applied sciences. Next I had a gap year where I had a fulltime job and studied mathematics B in the evening hours. After completing a state exam in mathematics B, I was able to enrol in the pre-master and later the master Industrial Engineering & Management. Here I did my specialization in “Production & Logistics Management”, where I could follow my interests in optimization of production environments. Overall it was quite a journey and I am glad I can finally say that, at least for now, I am done studying.

I would like to thank Simacan for the opportunity to perform my masters research at their company. In particular, I would like to thank my company supervisor Jeroen Soesbergen who helped me during my research and always had good feedback and ideas. I would also like to thank team Explorers for their nice conversations and help during my time at the office.

Furthermore I would like to thank my supervisors at the University Eduardo Lalla and Martijn Mes, for helping me out and guiding me through the research. Even though I had a little rough start and had to start over, they helped me finding a new subject and guide me along the way.

Lastly, I would like to thank my family, friends and girlfriend who helped me during my pre-master at the university and master at home during Covid times. Especially my fellow student and friends Matthew, Stan, Ivo and Rob, with whom I spend day after day online attending classes, working on assignments and not to forget spend lots of time in the beautiful city of Verdansk.

I hope you will enjoy reading this master thesis.

Michiel Mol
September 2022

Management summary

This research is conducted at Simacan, a Dutch software company that delivers software as a service. Simacan is the leading company in real-time monitoring of transport & logistics within the Netherlands. With all the data that is received from traffic and customers using the platform, Simacan is able to do more than only giving a clear overview of assets and expected arrival times. This research is focused on a problem that has developed in companies providing home-delivery services in recent years.

Over the past 10 years the use of methods for buying groceries online and having it delivered at home, experienced tremendous growth. The biggest and unexpected growth, occurred in the last two years due to the pandemic. The Covid-19 virus took over the world and people got to experience curfews, stay home obligations and quarantines. This has led to more and more people ordering their groceries through online methods to comply with regulations and avoid potential contamination risks. For home-delivery companies (HDCs) this meant almost a doubling in customers in only a year time. This brought a lot of pressure on their delivery plans. Due to the traffic reduction within that time, HDCs managed to keep the deliveries within reasonable boundaries. However, as the Covid-19 pandemic comes to an end and people are used to delivering online, current plans used by HDCs are in need of improvement. Traffic is rising back to its original level and is most likely continuing its growth as before the pandemic. Using the same scheduling methods, this results in very high chances of customer deliveries running late. In the daily operation, the HDC already asks its delivery drivers to maximise on-time deliveries to prevent unhappy customers. The HDC therefore asked Simacan to help them improve their plans before the execution (pre-trip) stage in order to maximise an on-time delivery.

To help Simacan, we performed research on how to improve the plans of HDCs. The goal of this research is to develop a method to improve current home delivery plans, making them more robust to withstand traffic fluctuations and maximise the likelihood of on-time delivery. We started the research by investigating existing problems in routing and scheduling within the home delivery sector. We identified that the rescheduling problem we are facing is called the vehicle routing problem with time windows (VRPTW). The inclusion of a robustness factor within scheduling, is a topic that has only come up in the last few years within the literature and is therefore a valuable research subject.

The method we have devised is one that, based on historical data, improves the current plan by optimising it on robustness through rearranging the stop sequence. We defined Robustness as the degree to which a driver is not able to deliver within the time window of the customer. After analysing 4 years of historical data, we were able to create probability functions that represent the probability of a late delivery with regards to amount of minutes planned before the end of the time window of the customer. Because the HDC has hubs spread around the Netherlands and situated in different rural and urban areas, they all have their own probability pattern. To overcome this, we created separate functions for each hub. We

subsequently used the probability functions to create a robustness function to optimise a trip based on this probability. This robustness function consists of four elements and is based on the method from the paper of Hsu et al.(2007). Figure A gives a visual representation of the function. We determined a penalty robustness value of “M" for planning before and after the time window, and start giving a penalty at a certain point “s” within the time window where the probability of a late delivery starts increasing before the end of the time window.

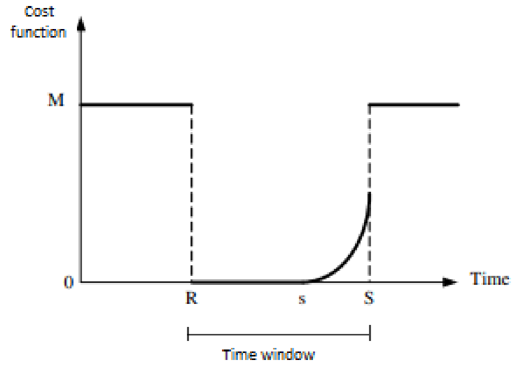


Figure A: Visualisation of the Robustness function

In order to optimise a single trip, we created a method that uses a tabu search algorithm with a 2-opt operator. The algorithm continues to search the solution space for better solutions in order to optimise a single trip based on the robustness value which follows from our function. To calculate the robustness value of an entire trip, we decided to add up all individual robustness values of the customer level to a total robustness value that represents the robustness of the trip. This means, the lower the outcome of the robustness value, the better robust solution we get for the plan of a trip. To get the routing and travel time of the new emerging trips after rescheduling, we used different APIs from Simacan to determine optimal routes and travel time.

To validate our method, we performed multiple experiments and used several different case studies for testing our approach. Our experimental data set is part of an original plan and consists of 118 trips with different characteristics to be able to cover most of the different scenarios. These experiments showed us some relevant information about the plan of the HDC and its potential of improvement. We found out that the current plan is primarily based on optimising the distance while delivering within the time window. This confirms the outcome from our data analysis and the information we received from Simacan internally. Furthermore, we confirmed the large potential for robustness improvement of current plans. From the parameter optimising we concluded that robustness could be improved by around 30%, without looking at distance increase.

After performing the parameter optimising experiment, we performed several others extending and testing the basic robustness method. These experiments included: a composite penalty function, alternative starting solutions for running the algorithm, and lastly combining robustness and distance optimisation within one multi-objective function. We concluded that the alternative penalty function did not give significant different results for our instance, and need future research to investigate the potential added value. Using alternative starting solutions showed us the capability of our algorithm to find good near optimal solutions. Starting to optimise with different plans did not give significant better results and needed consistently more time to solve. Finally, we found some valuable information about optimising on multiple objectives. Using different weights that can be set in advance to the distance and robustness objective, the algorithm was able to find valuable improvements. By increasing the travel distance by only 4% the algorithm was able to find significant robustness improvements of at least 20%. Benefits of these improvements are higher customer satisfaction, as they should get less late deliveries, and more grip and insight on the daily operation.

From our research, we conclude that the robustness increase of home delivery plans based on historical traffic data has added value to the attended home delivery problem. When using our created method on a home delivery plan of the HDC, a significant potential increase in robustness can be made. However, only to a certain extend they are useful within a real life application. From our experiments we determined that when improving on robustness alone, this does not outweigh the increase in travel distance. Using a multi-objective function shows far more potential for this case and should therefore be further investigated. We therefore recommend to perform further research on these weights in order to find a good balance. Next to that, the current running time of the algorithm is not very fast and will not suffice for the current daily plan. Some code optimisation is needed in order to use the method for real-life optimisation cases. Also, we scoped out the use of breaks within the rescheduling of trips. When Simacan is able to detect breaks within the execution of plans, it is recommended to perform research on how to implement breaks within the rescheduling algorithm to further increase the feasibility of the revised plan. Lastly, we believe that the improvement based on historical data becomes outdated as traffic evolves over the years. Therefore the distribution functions of all hubs need to be updated frequently in order to retain the improvement potential.

Table of Contents

List of Figures

List of Tables

Abbreviations

AHDP	Attended home delivery problem
HDC	Home Delivery Company
API	Application programming interface
CCP	Chance-constrained programming
CNI	Consecutive non-increasing iterations
DC	Distribution center
ETA	Expected time of arrival
KPI	Key performance indicator
MOO	Multi-objective optimization
NP-hard	Nondeterministic polynomial time **-** A problem is NP-hard if an algorithm for solving it can be translated into one for solving any NP-problem (nondeterministic polynomial time). NP-hard therefore means “at least as hard as any NP-problem”.
R-MOEA	Robust multi-objective evolutionary algorithm
SaaS	Software as a service
SCT	Simacan control tower
TS	Tabu Search
TSP	Traveling salesman problem
TSPTW	Traveling salesman problem with time windows
VNS	Variable neighborhood search
VRP	Vehicle routing problem
VRPTW	Vehicle routing problem with time windows

Glossary

(Customer) Stop	When referring to a (customer) stop, this represents a single customer location where a delivery need to take place.
Route	When referring to a route, this represents the road to be taken between two locations. For example between two customers or a depot and customer.
Time window	A time window is a period of time, consisting of a start and end point, in which a delivery should take place.
Travel time	The time it takes to travel from point A to point B.
Trip	When referring to a trip, this represents the entire routing sequence between a series of customers and the depot.

1. Introduction

The first chapter of this thesis starts with an introduction of the company where the research is taking place (Section 1.1). After that follows a research motivation in Section 1.2, whereafter in Section 1.3 the reader will be introduced to the problem that will be addressed. After the problem introduction, in Section 1.4 the research questions will be elaborated that will help finding a solution to the problem. The chapter concludes with the research scope in Section 1.5 and the research design in Section 1.6.

1.1. Company description

Simacan is a SaaS (Software as a Service) company that was founded in 2013. It is based in Amersfoort and currently employs more than 75 people. Simacan offers an open and vendor-independent cloud platform for digital cooperation in transport & logistics. Simacan serves a large portion of the Dutch retail market including the leading supermarket chains, e-tailers and postal-parcel companies, processing more than 25 million deliveries on a yearly basis. Simacan enables fast and secure digital cooperation with – and between – transport companies and shippers to tackle industry-wide challenges, such as reducing transportation costs, optimizing the transport supply chain and reducing CO2.

To be able to do this, Simacan developed the Simacan Control Tower (SCT) illustrated in Figure 1. The SCT is a cloud-based software platform where customers will get unambiguous, real-time information about their shipments.

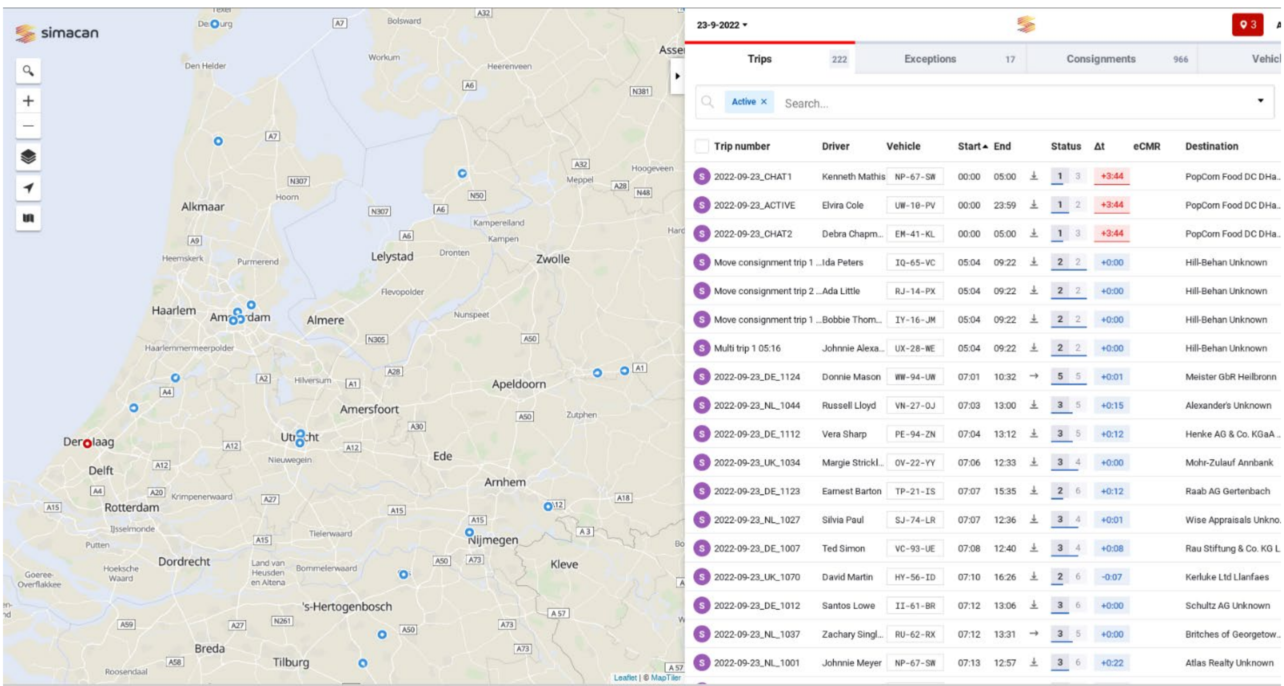


Figure 1: Preview of Simacan Control Tower interface

The platform consists of multiple elements and interfaces to ensure that all players in the supply chain are simultaneously involved and informed in various ways. Simacan calculates the fastest and safest route with traffic, last-mile delivery instructions and in-cabin information whilst considering traffic accidents, congestion and transport regulations to avoid losing time at the dock or next hub. The platform offers clear graphs, overviews, and effective data filters to help customers track and analyse their shipments on a real-time basis.

A difficult part of this service is that each company has its process and way of working. Therefore, Simacan differentiates between three different kinds of sectors, namely supply chain (transport/retail distribution), parcel (transport from e-commerce to hub) and home delivery (from a store, hub or distribution centre). Each sector has its way of transportation.

Each party within the process of supply chain and home delivery has different information needs. Therefore, various interfaces have been developed to be able to serve all sectors and create easy to use transportation overviews for each stakeholder in the supply chain. The four main applications are listed below:

1. **The Simacan Control Tower (SCT):** Within the control tower all the magic happens. It gives the overall map as shown in Figure 1, where one can monitor all trips and track vehicles from the begin to the end of the supply chain.
2. **Store displays:** A screen within stores that show the arrival time and load information of trucks.
3. **Arrival displays:** A screen that provides insight into times and peaks of arriving freight traffic at unloading addresses (Hubs/DCs).
4. **End customer notification:** For customers with home delivery there is a notification service via e-mail, SMS or a customer-specific environment that informs consumers at home at what time the delivery driver will arrive.

With the help of these four applications, Simacan has earned her place in the Dutch retail market and is heading into Europe in order to expand their business and help more and more companies to create a sustainable future for logistic transport.

1.2. Research motivation

Currently Simacan offers a platform that helps to eliminate the complexity of the execution of transportation. With all the data that is gathered from customers using the platform, Simacan is able to do more than giving a clear overview. This data can also be used to help customers in other ways.

On an everyday basis, shipments deviate from created plans within transportation companies. With the help of gathered data from Simacan's platform, it is possible to compare the initial day-to-day plan with what really happened in practice. The realised data of different transports from for example distribution centers to stores or from hubs to customers can be analysed and compared on different levels, like differences between planned and realised trips, planned and realised routes, differences between truck drivers, and overviews of loading and unloading times.

In this wealth of data, potential is hidden to provide quick insight for (potential) customers on what improvements in their logistics operations are feasible. Unlocking these quick insights is yet an unexplored territory for Simacan.

1.3. Problem description

The data of Simacan is widely used for different innovation projects within the transportation sector to help customers with their logistics operation. One of these projects is using traffic history and prediction to determine expected arrival times (ETA's), as traffic is a challenging subject. Namely, in the last 20 years a significant change within modern day traffic emerged. The increase in passenger cars only is already 14% (CBS 2021) , let alone transport vehicles, whose travelled distance grew the most over the last 5 years (CBS 2020). Over the last 5 years the growth in transportation vans and heavy goods vehicles was respectively 14.7% and 12.6%. According to the yearly traffic report of Rijkswaterstaat (Rijkswaterstaat 2019), the traffic intensity between 2000 and 2018 increased with about 28% (Figure 2). This increase in traffic has a large influence on traffic jams. Also more maintenance on roads is needed. For many logistics companies this will create a growing challenge as it will have a significant effect on timeliness and on-time deliveries. Specifically looking at the home-delivery companies who are dealing with customer specific time windows and tight plans. Next to that, as can be seen in Figure 3, the influence of the Covid-19 virus nearly doubled the need for home delivery within the Dutch retail-grocery market (Simacan 2021). To avoid crowded places and human contact, people are choosing home-delivery over going to the supermarket themselves. Therefore, well-designed delivery plans are crucial to fulfil customer expectations for the home-delivery sector. Also from the business perspective the HDC wants to cope with this increasing demand. Hence delivery plans need to be robust and able to

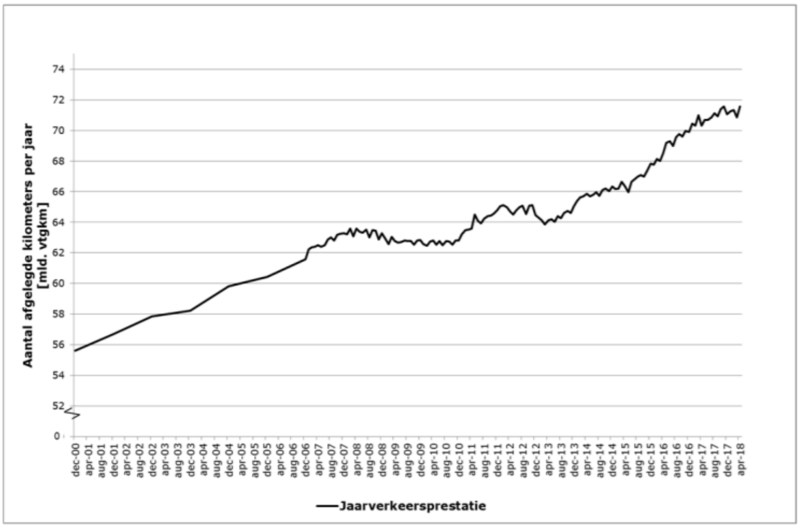


Figure 2: Traffic intensity from dec 2000 until April 2018 by Rijkswaterstaat (2019)

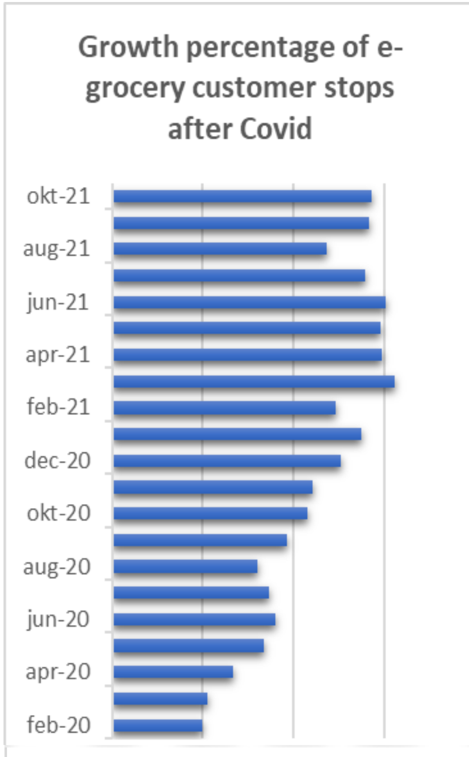


Figure 1: Home-delivery growth after the Covid virus, based on data from Simacan

withstand traffic fluctuations. This can be done beforehand in the pre-trip stage as well as during the execution in the on-trip phase.

The part of improving plans in the on-trip phase has previously been researched by a fellow master student in his paper about "Dynamic rerouting for an optimal delivery strategy" (Scholten 2021). Thus, this research will be focused on the pre-trip stage. The pre-trip stage is defined as the stage when the delivery plan is received, but the execution still needs to take place. The incoming plan of customer deliveries will be analysed and modified to increase the robustness and customer satisfaction of the initial plan, where robustness is defined as the degree to which the plan can withstand traffic and other delays within a specific trip. This can be managed by delivering as little as possible near the end of the customers time window. This will increase the probability of a trip being delivered on-time. A more specific meaning of robustness will further be specified in the literature research.

1.4. Research questions

As mentioned in the previous section, creating delivery plans for home-delivery companies is becoming more and more a challenge with modern-day traffic. These plans need to be robust to be able to withstand the heavy traffic increase that emerged over the past years. Using realised data from Simacan, it is possible to analyse the past and use this to improve current plans. The goal of this research is therefore formulated as the following main research question:

How and to what extent can current plans from home-delivery companies be improved (before execution/pre-trip) in terms of robustness?

To find a solution for the main research question, we break it down into smaller subjects, of which each contributes to answering a different section of the main question. They also provide guidance for the research. The following sub-questions will be addressed:

1. *What do current plans from home-delivery companies within Simacan look like? What is the current situation and what KPIs are involved?*

We aim to create a method in order to improve plans of home-delivery customers of Simacan in terms of robustness. In order to do so, we first need to know more about the current situation. A context analysis is needed to know what current plans look like, what KPI's are used and how is the current situation.

After performing a context analysis of the problem to get more insights, we need to conduct a literature research to obtain knowledge. The first literature question we will try to answer will be about routing and scheduling for home-delivery companies and how this is addressed within the literature.

2. *What does the literature contain about improving delivery plans for home-delivery?*

The second question in the literature research will be about robustness. To be able to make more robust plans, we need to know what the specific definition is of robustness within literature, how this is measured and what strategies exists in order to improve this in plans for home-delivery companies.

3. *How is planning robustness defined in literature and what are known strategies to measure and improve robustness?*

After the literature phase, we need to consider how the found method can best be used in building the solution for improving the robustness of home-delivery plans. This insight consists of assumptions, requirements, and other necessary information to build the solution method. This leads to the third research question:

4. *What robustness improvement method can best be applied to the Home Delivery Company case study?*

After the implementation phase has been completed and new more robust plans can be created, the assessment of the solution approach and its performance need to be validated. Here we need to perform experiments on different trip instances in order to conclude the performance of our new plan for different scenarios in comparison to the current plan.

5. *How does the solution method perform under different experimental scenarios?*

1.5. Research scope and limitations

This research takes place at Simacan in Amersfoort. The topic of planning optimisation is complex and can be extended in many ways. Therefore, before the research is conducted, some boundaries and size of the study need to be defined. The scope of this research and limitations of certain choices are as follows:

Scope:

- The focus of this study is to create a solution method to optimise existing plans. The scope for this solution method is limited to planning and routing. The method will be limited to three steps: a threshold that will be used to determine which orders in a trip need to be replanned, a search space where the corresponding order can be replanned to, and the final replanned solution.
- To reduce the complexity of the research and due to a lack of information, we will restrict the problem in a couple of ways. First, we do not include any financial consequences of the optimised plan. Some examples of these financial consequences are the costs for increasing travel distance and extra working hours. Next to that, we will only look at single trip mutations because important data like capacity allocation is unknown. This means that we will not swap

customer stops between different trips but will only look at optimising a single trip by rearranging the order.

- As customers of Simacan are from different sectors we will focus on one sector in particular. The sector that will be addressed in this research is the e-grocery sector. In this sector we will specifically focus on the Home Delivery Company (HDC).
- The data that will be used within this research will mainly come from the databases of Simacan. In agreement with customers, it may occur that their data will also be used in order to create a combination with data from Simacan.
- In order to get a good understanding of feasibility and travel times of the newly created routes from the methodology, we will use the existing travel time and routing API created by Simacan which includes traffic profiles and shortest path routing.

Limitations:

- Scheduling for home-delivery companies is not done by Simacan itself. Customers deliver their plans to Simacan created by their own planning instance. This research is conducted to adapt these given plans to be able to make them more robust. The plan for all customer orders will therefore not be recreated to trips with corresponding routes from scratch.
- The solution method that will be created and tested will be based on the operation of the HDC. As not every customer of Simacan works according to the same operation, the methodology needs to be adapted when used for other customers.
- Data used in this research will also include data from the past two years. Due to Covid-19, which became a pandemic virus in March 2020, this data is very different from the years before. As people where obligated to work from home as much as possible, there was not as much traffic as in the previous years. Also, we do not know how this will affect future traffic and if we will get back to the old situation before the virus. Therefore, the inclusion of this data will have a significant influence on the solution that will be created.

1.6. Research design

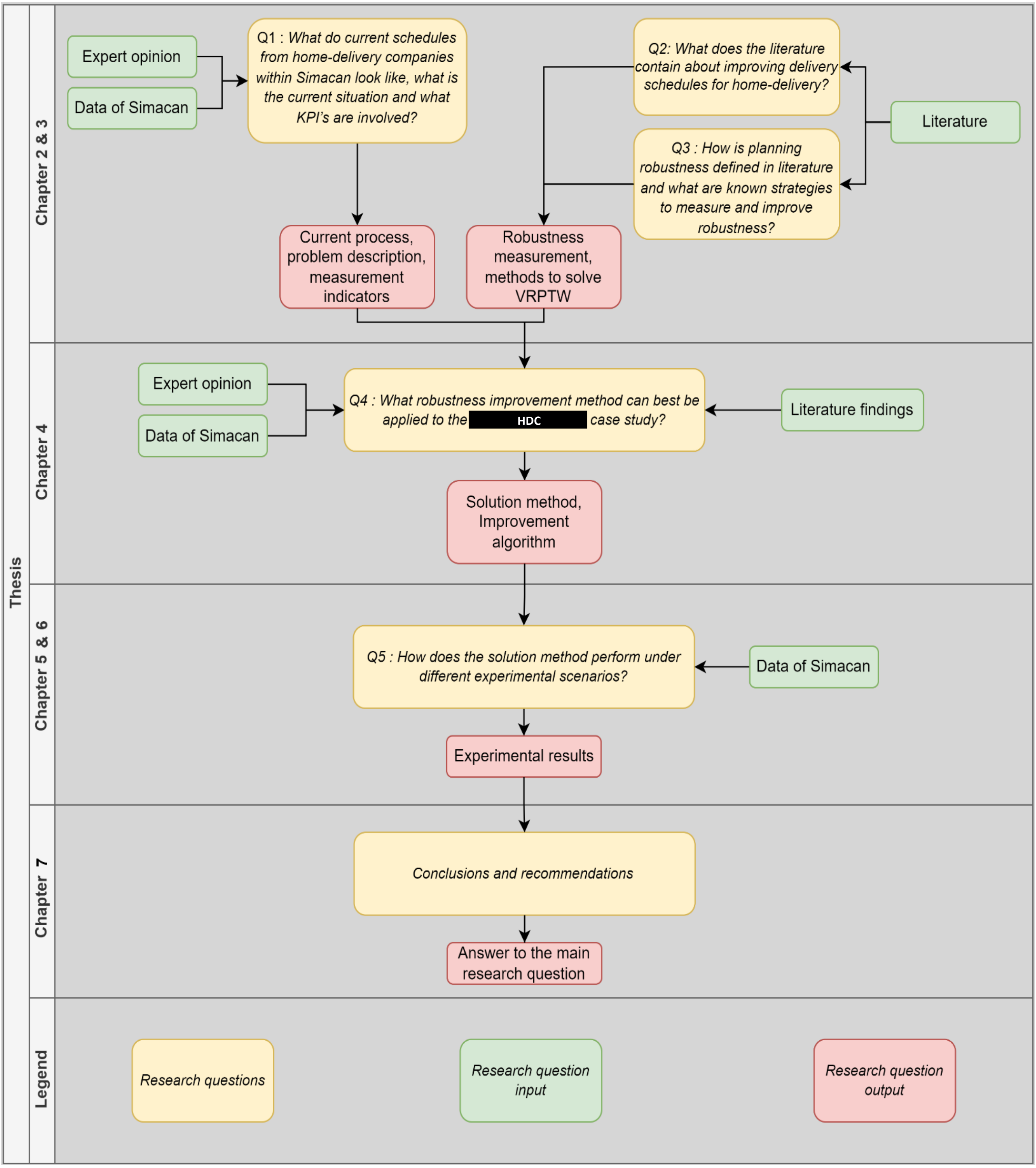


Figure 4: Research design

Figure 4 shows the relation between the research questions (yellow), what input is needed for each research question (green) and what output is generated by each research question (red). It also shows the chapters in which the research questions are answered.

2. Problem context

In this chapter, a detailed description of the problem will be provided. Here we give an answer to the first research question. We start in Section 2.1 with an introduction about home delivery plans. Next, we elaborate on the traffic development in Section 2.2. Then, the severity of the situation of lateness within the plan of the HDC is reviewed in Section 2.3. In Section 2.4 we give insight into mutual differences between delivery hubs. Finally, in Section 2.5 and 2.6 we focus on how Covid-19 influenced the situation of the HDC and their current solution.

2.1. Home delivery plans

As introduced in Chapter 1, Simacan created the SCT to visualise and calculate the ETA of delivery routes for their customers. These delivery routes are based on plans created by clients of Simacan and consist of multiple customer stops with each their own characteristics. In this section, we explain the process of how these home delivery plans are created and how this is visualised by Simacan in the SCT.

The process of scheduling starts with a customer that needs groceries. The customer then goes to the app or webpage of the specific retailer and orders the needed groceries. At the checkout, the customer is offered different time slots in which the delivery can take place. We specify the order period for a specific day as [0, T], where 0 is the first available moment in time for which the order can be made, and T the so-called "Cut-off time", which will be the last moment the customer can place an order for the day. After this time T, the plan for a day will be created by the planning department of the retailer. This daily set of customers with different destinations, grocery volume and time windows are becoming a difficult problem to solve. They need to be combined into different routes matching their time windows and vehicle capacities. This problem is part of a so-called Vehicle routing problem (VRP). This problem emerges when one has a fleet of vehicles that need to deliver goods to a given set of customers by a set of routes. Many adaptations on the VRP exist because most of these problems have their own specific constraints, which includes characteristics such as vehicle limitations, cost controls, time windows, resource limitations concerning the loading process at the depot, and many more. For this specific problem for home delivery we also have certain characteristics that have to be taken into account when creating viable routes and plans. Because we are dealing with food that in most cases can perish, the customer needs to be home when the delivery takes place in order to store it properly afterwards. Therefore the customer indicates a time window consisting of several hours during which they are available for service. These time windows are in general fixed and cannot be changed, we call these "hard time-windows". These hard time-window constraints increase the complexity of determining optimal delivery routing. However in most cases we assume that people tend to be home a small period before and after their given time window. When a delivery arrives early, or with an acceptable delay, it can still take place but with a certain penalty. The process from start to end from a customer perspective is shown in Figure 5. The characteristics of this specific home delivery problem point to the Vehicle routing problem with time windows or VRPTW. This VRPTW includes different time-slots, locations and delivery service times (the time it takes for the delivery driver to hand over the order at the customer stop) based on the volume of the orders. The VRPTW will be further elaborated in the literature research. The VRPTW can be solved by various heuristics and metaheuristics to create a feasible plan and routes for the day.

Figure 5: overview of the e-grocery process from a customer perspective

When the routing and plan is created, Simacan receives the plan a couple of hours before execution. Experience of Simacan shows that these plans are optimised based on travelling the least amount of distance. The plan is provided with all the necessary trip and stop information with corresponding arrival and departure times, time windows and delivery load. The plan information is converted to a structured overview within the SCT. With the use of an Application Programming Interface (API) the stops are connected with the shortest route and visualised within the SCT. From the overview shown in Figure 1 in Section 1.1, we can open a trip to get further information.

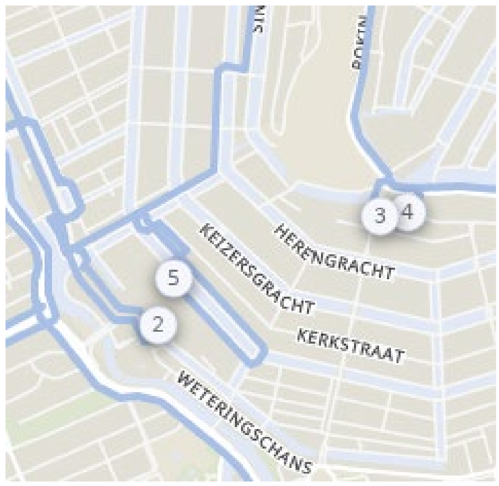


Figure 6: Overview of customer trip

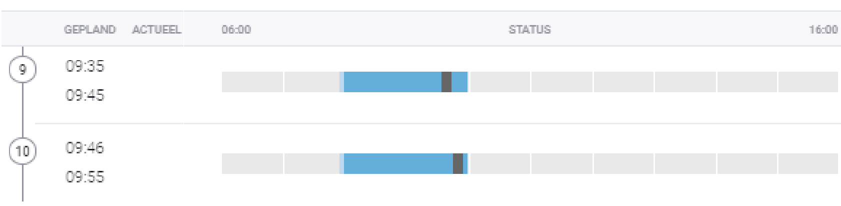


Figure 7: Detailed information of customer trips

Here, we see a visualisation of the trip with associated routes and corresponding stop sequence as shown in Figure 6. In order to give a clear overview of the customer order, delivery times and load, one also get more information of the specific stops as shown in Figure 7. When the trip has started, real-time information like expected time of arrivals (ETA's), realised delivery times and vehicle location are shown in this overview.

2.2. Traffic delays

After introducing the traffic situation of the past years in Section 1.3, we will give insight into the kind of impact this has on the delivery situation of the HDC. As mentioned, the increase and fluctuations in traffic are difficult to manage. Especially for the home delivery sector where timeliness is a significant part of their operation. Due to the fast growth of this sector since February 2020, when Covid-19 became a pandemic virus, home delivery companies experienced rapid growth and became more and more dependent on traffic. Until now the sector managed to adapt to this growth due to the decrease in traffic that started at the same time.

In order to give some insight, Figure 8 shows, based on Simacan's data, the amount of traffic delay of the past four years. This delay is measured by taking the Free-flow conditions as the starting point. The free-flow condition is the travel time when one can drive freely without any traffic delays. As can be observed, traffic delay decreased significantly when Covid-19 became a pandemic at the end of February 2020. For home delivery companies this has increased on-time deliveries on their tight plans as there are fewer fluctuations in travel time caused by delays. However, it is unclear how the current traffic situation will evolve in the future and if it will return to the previous level and continue its growth from recent years. Therefore plans are in need of improvement and adaptation to the future situation.

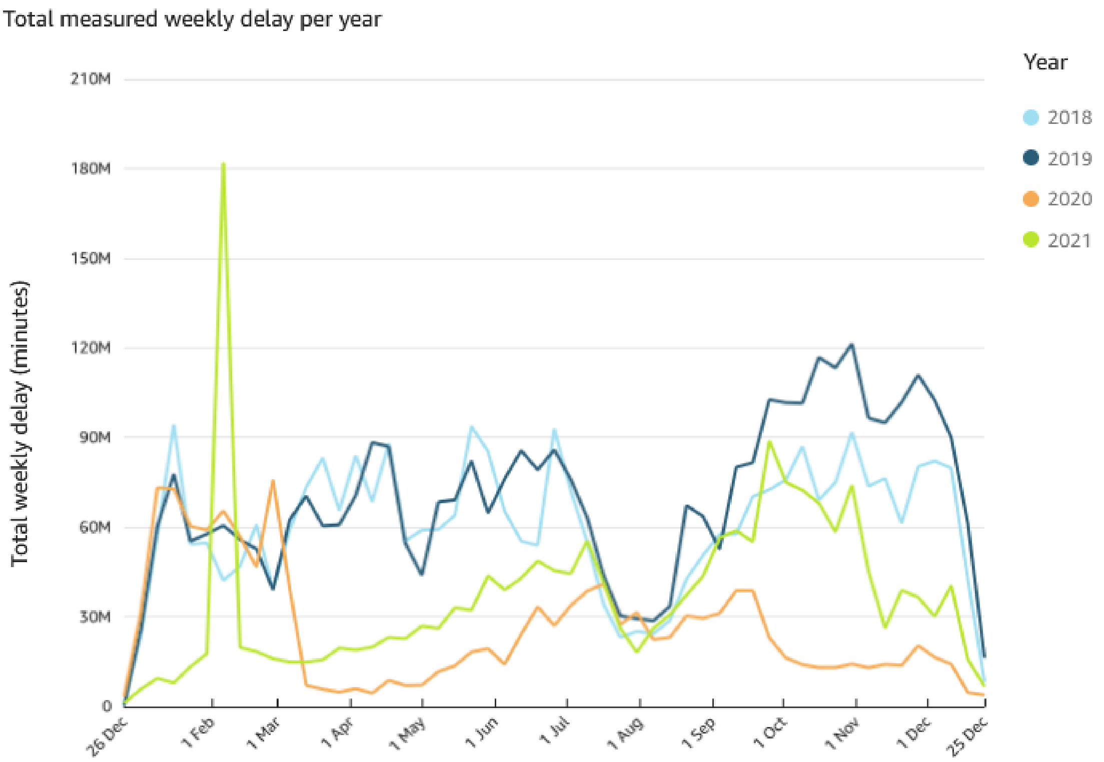


Figure 8: Weekly traffic delay per year in million minutes

2.3. Severity of the situation

In order to get more detailed information about the impact of the traffic situation and customer growth on the current home delivery operation, we conducted a small research based on customer data of the HDC. What has been noticed in earlier research of Scholten (2021), is that plans from the HDC are not very robust in their ability to handle any possible delay. A KPI used by the HDC to measure robustness is the percentage of stops that are delivered within the time window. As can be seen from Figure 6 in Section 2.1, customer deliveries are planned close to the end of the customers time window. Scholten noticed that this is happening very often within the plans of the HDC. The downside of this occurrence is that any resulting delay within a trip is very difficult to make up for. As opposed to other sectors where you might be able to work twice as hard, it is not possible in the transportation sector to drive twice as fast to make up for any lost time. The consequence of this is that even small delays in the delivery process will have effect on all following stops and increase the possibility of delivering outside the time window. In Figure 9 we can see an example of what is happening a lot at the HDC. Here, stop number 7 is planned not very far from the end of the time window of the customer (the black line). By doing so, the risk of not delivering within the time window becomes high. As we also see, this happened for this stop. Due to a delay earlier in the trip the driver has not managed to deliver in time.

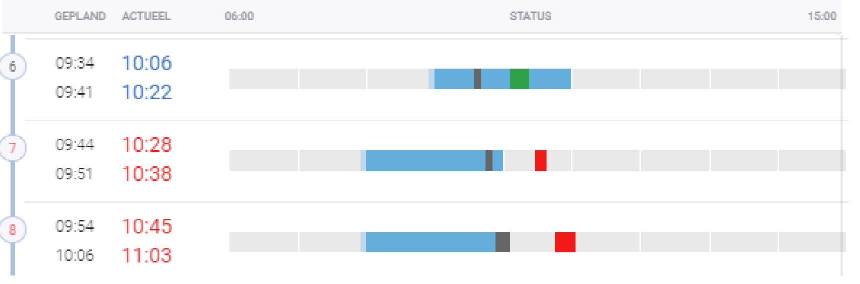


Figure 9: Example of delivery outside the time window caused by traffic delay and end of time window planning

To get to know the impact of the described situation, we analysed data of the past four years to investigate the current way of planning. With available customer data of the HDC, we determined for every month, the number of stops that were realised outside the time window. We created two graphs, combined all months and selected stops that were planned to be delivered within 10 and within 15 minutes to the end of the time window. These graphs can be seen in Figure 10. In these graphs, we see that the probability of not meeting the given time window becomes larger when the stop is planned closer to the end of the time window. Furthermore, we can see that the number of stops delivered late is decreasing over the past two years.

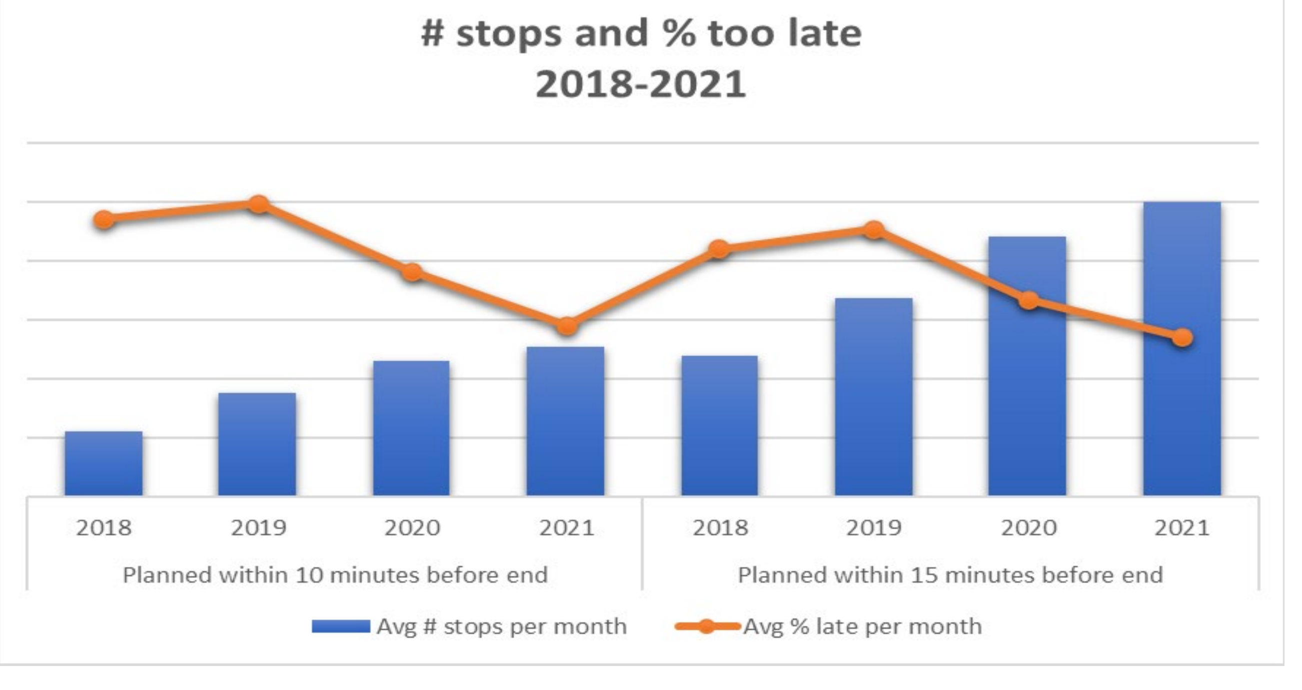


Figure 10: Number of stops and percentage late, based on data of HDC from 2018-2021

Multiple reasons can be found from this. However, one of the main reasons we expect to be the decrease in traffic caused by Covid-19. Since the traffic reduction caused by Covid-19 is likely to disappear in the coming years, we expect these percentages to increase again. In order to prevent this, we need to improve the current way of scheduling and create more robust plans that are able to withstand traffic delays better.

As mentioned before, the HDC is a large company in the home delivery sector. From just data, as shown in Figure 10, it is hard to determine the exact size of the problem. Therefore, to provide a better understanding of the scale of the HDC, we gathered a year of data (2021) and looked at an average daily planning operation. Figure 11 gives some information about the quantities of an average day at the HDC. As shown, every day a large amount of customers need to be planned, with each their own time window and location. This results in many trips driven by several vehicles divided over multiple hubs. This is a very large logistical problem which need to be solved every day.

Figure 11: Overview of an average daily operation of the HDC.

2.4. Delivery hubs

The HDC delivers their groceries all over the Netherlands. This is done from different distribution hubs that are spread over the country in order to be able to service every customer. Earlier research within Simacan has shown that there are substantial differences between the timeliness of these hubs. The research, based on the hubs of the HDC, revealed a time differences between planned and realised of up to 10 min over several hubs. A selection of the outcome can be found in Figure 12. Here we see on the y-axis a set of different hubs and on the x-axis we see the median percentage error (in minutes) over all trips from that specific hub. From the figure we can conclude that the average error for the hubs deviate between 1 and 10 minutes. One of the main reasons for this difference can be explained by the fact that some regions around hubs are more densely populated than others. This leads to more traffic delays for certain hubs located, for example, in the Randstad and hubs with less traffic located near rural areas. As the lateness distribution can therefore be different for a region, this also needs to be taken into consideration when trying to improve the robustness of plans. Therefore we also took this into consideration when investigating the percentage of stops delivered late.

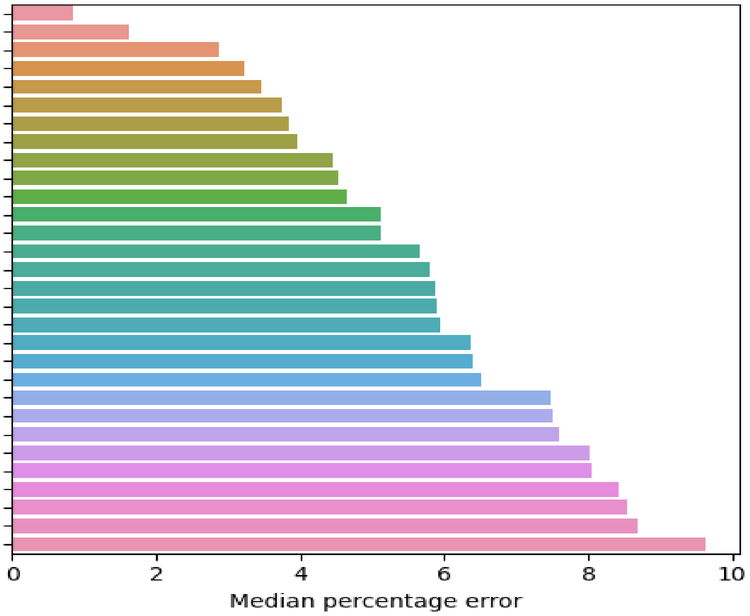


Figure 12: Insight in timeliness difference of hubs, based on research of Simacan

In Figure 13, we see our findings of the current lateness situation of all hubs combined. This lateness is plotted against the amount of time (in minutes) a stop is planned before the end of the time window. The figure consists of two graphs. The orange bars are representing the percentage of customer deliveries that are delivered outside the given time window, when planned that amount of time before the end of the time window. The blue line represents the corresponding amount of customer deliveries on that specific time (on time and late). Both graphs are based on the same x-axis, which represents the amount of time (in minutes)

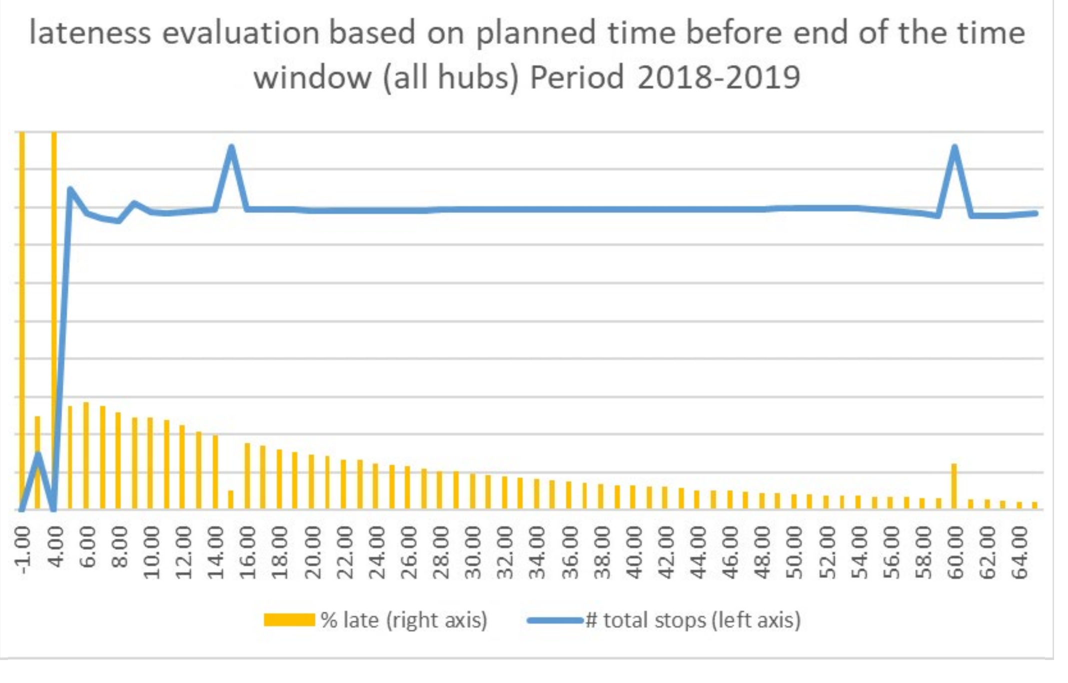


Figure 13: Graphical visualisation of lateness percentage based on planned before the end of the given time window (data 2018-2019)

the delivery is planned before the end of the given delivery time window. As can be seen from the figure, there is a smooth degrading slope on the percentage delivered outside of the time window (late). However we see two exceptions on the 15 and 60 minute mark, which are more common time units used in their plan. The reason behind these outliers is still unknown.

When we further investigate the data and divide the data into a hub specific view, where all trips that start from a single hub are fit into the same graph, we see the same degrading effect for almost all existing hubs. However, when comparing the hubs we see heavy fluctuations in the starting percentages for the top of the slope. This confirms earlier research of Simacan that there are indeed significant differences between hubs. Figure 14 gives an overview of these differences per hub, specified to first couple minutes planned before the end of the given time window. We took the lateness percentages based on the first 4 to 7 minutes planned before the end of the time window. The reason behind this is that there was a certain fluctuation between the hubs at how many minutes before the end they would start to plan. On the x-axis we have the different hubs. The blue bars represent the starting percentage for the top of the slope. The orange dots are representing the corresponding total amount of stops (on time and late) at this starting point (based on a logarithmic scale).

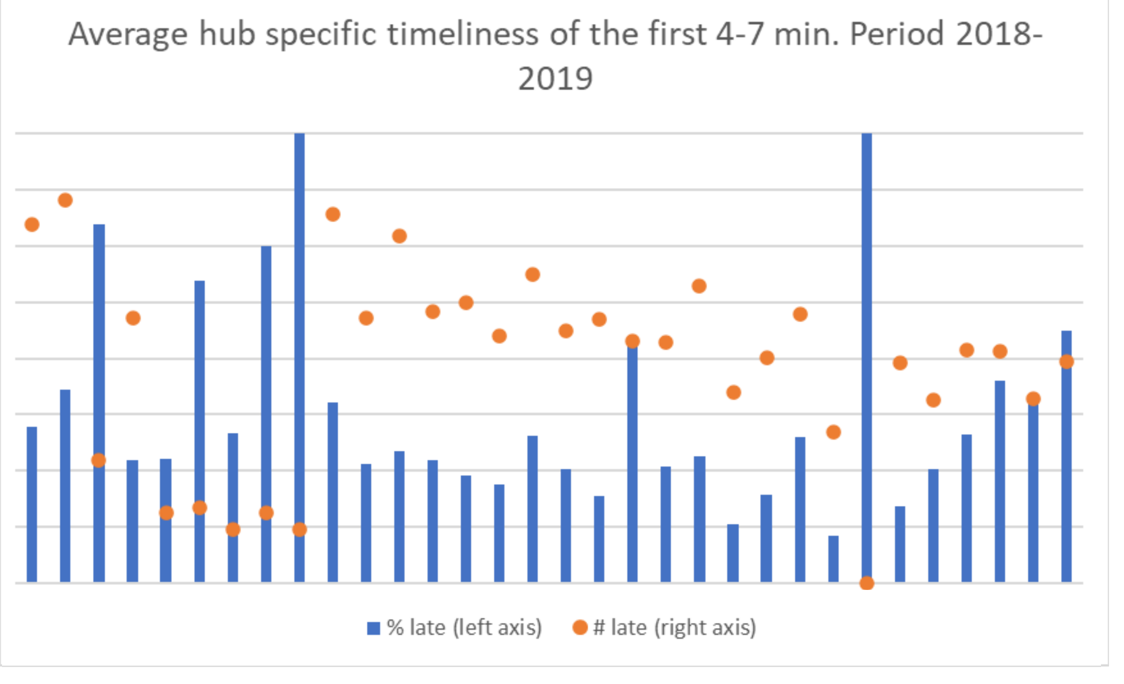


Figure 14: Hub specific timeliness based on planned deliveries close to the end of the time window (data 2018-2019)

2.5. Influence of Covid-19 on the data

When we analyse data in order to get information about previous years, we see a large difference emerging with regards to traffic. In Section 1.3, we discussed this decrease in traffic. With the arrival of the Covid-19 virus in February 2020, traffic intensity dropped to the same level as it was back in 2003. Due to the fact that people were forced to stay at home during lockdowns or quarantines, more and more people started to order their groceries online. For the timeliness as shown in Figure 13 and Figure 14, this had a quite significant impact. For the HDC this meant a huge growth for their customer base. When we update Figure 13 with data of 2020-2021 we get the new graph as shown in Figure 15. Here we see that in the past two years some stops are even planned outside their given time window, noted by the minus x-axis values. The HDC was not able to plan these orders within the given time due to this customer growth. On the other hand, for trips that are planned within their time window, we see almost the same degradational slope starting at time 0 as we encountered in the data of 2018-2019 in Figure 13. However, due to the decrease in traffic the starting point of this slope starts now at about 20%, which is 10% lower than before. This means deliveries are less late than before, while the amount of trips have increased. As expected, when the trip is planned outside the time window we see a much higher percentage of stops that are delivered late.

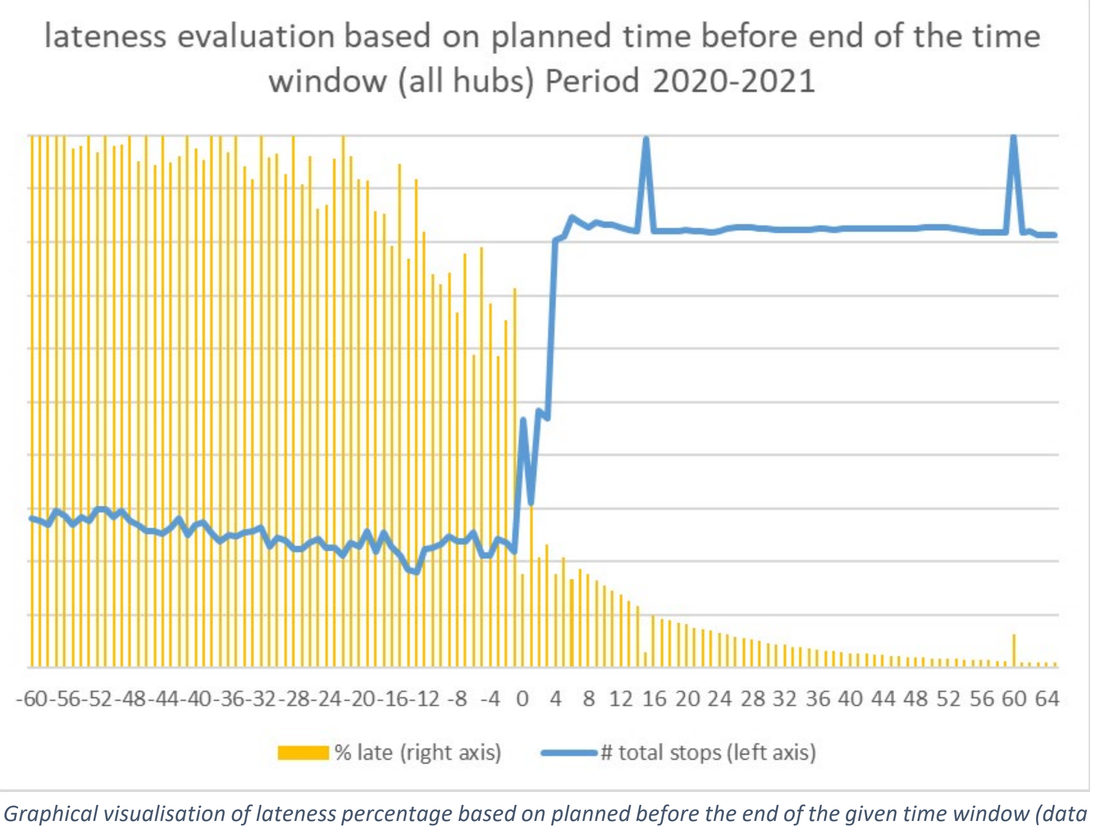


Figure 15: Graphical visualisation of lateness percentage based on planned before the end of the given time window (data 2020-2021)

When we also update data of Figure 14 to 2020-2021 visualised in Figure 16, we see that we have some more hubs on the x-axis. Overall, there are quite some noticeable different starting percentages for each hub. However, these percentages are overall lower than the 2018-2019 period. As the difference between the two periods is quite significant, we need to be careful in how we adapt current plans based on this historical data. As it is unclear how the current traffic situation will evolve, we will have three possible scenarios. One scenario where we will expect the situation to evolve into how it was in 2018-2019. Secondly a scenario where we expect the current Covid-19 situation to continue and build further on the data of 2020-2021. Lastly, a hybrid situation where we expect the current situation of 2020-2021 to continue but more and more evolve back to the situation of 2018-2019.

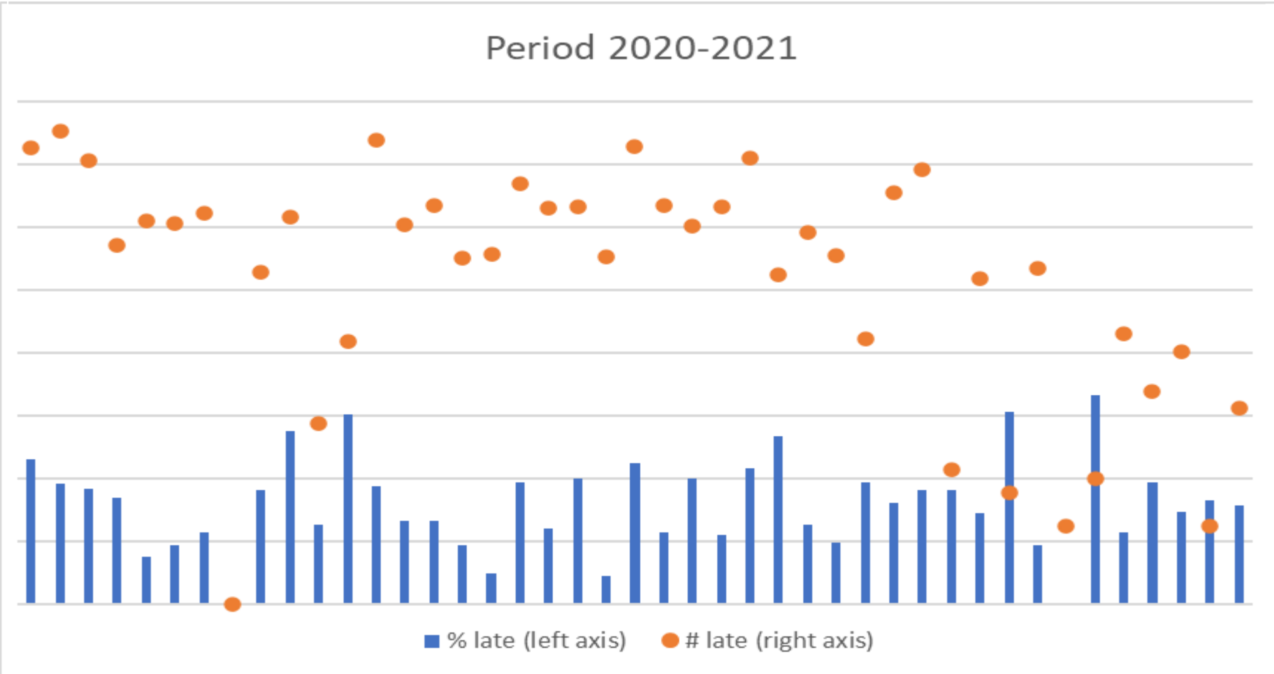


Figure 16: Hub specific timeliness based on planned deliveries close to the end of the time window (data 2020-2021)

2.6. Current solution by the HDC

The HDC is also aware of the rising problem that decreases their KPI's of on-time deliveries. The current planning method, which they are using, is primarily focused on driving the least amount of distance, while the delivery still needs to be planned within the customers time window as much as possible. As shown in Section 2.1, this often leads to deliveries planned close to the end of the time window. Within the execution of the plan this leads to many deliveries running late due to any resulting delay within the delivery process. Some examples of common delays are: the employee is unable to find the given address, the customer is not at home, traffic delay somewhere in the trip and the trip does not start on time due to absence or lateness of the driver.

2.7. Conclusion

This context analysis describes the case of the Home Delivery Company (HDC). Customer arrivals follow a fixed path from determining what groceries to order, to giving up their address and selecting a time window in which the customer is home and able to receive the groceries. Until a certain moment customers can place their order for the next day, whereafter a plan will be created. Data shows that these delivery plans are resulting in a relative large portion of deliveries not being delivered within the given time window at the customer.

Over the past 20 years, traffic intensity increased about 28%. This had a large influence on the lateness of drivers. With the current insecurities around the Covid-19 virus, like the introduction of a lockdown, working from home recommendation and curfews, traffic intensity fluctuates to as low as 20 years ago. Therefore it is hard to predict what will happen on a normal day. Due to this growth we see more and more stops planned to be delivered near the end of customers time windows and some not even planned within. As the possibility of not delivering in time increases when stops are planned closer to the end of the time window, this has a negative effect on the overall robustness. A KPI used by the HDC to measure robustness is the percentage of trips that are delivered within the time window. Due to the decrease in traffic caused by Covid-19, these plans did not result in a higher percentage of deliveries outside the given time windows. However, it is unclear how the traffic situation will evolve. If traffic levels will return to their original level and keep on increasing as they did over the past 20 years, this will have large consequences as the sector nearly doubled in customers. In order to prevent more lateness, plans are in need of change to become more robust for the coming years. As we do not know how the situation will evolve we will need to consider solutions based on different traffic situations. There are three possible scenarios: a scenario based on old data from 2018-2019, a scenario based on current data of 2020-2021 and a scenario based on a hybrid solution where data from 2018-2020 will be taken into consideration.

3. Literature review

Within this chapter relevant literature about the subject of our research will be reviewed. This review aims to provide an overview of the existing methods that are used for similar or related problems. From here we determine the research gap that our research needs to fulfil and which of the existing methods can be used to help solve the problem. In this chapter, we try to find an answer to the second and third sub-questions of our research.

Section 3.1 starts by explaining the attended home delivery problem and how this problem is addressed in literature. Subsequently Section 3.2 gives a brief introduction of the vehicle routing problem (VRP) and some variants. In Section 3.3, we will go in-depth into time windows which lead to the vehicle routing problem with time windows (VRPTW) and the Traveling salesman problem with time windows (TSPTW). Section 3.4 will dive deeper into the most used methods to solve a VRP and specifically for the TSPTW. Finally Section 3.5 describes the definition of robustness within plans and dives deeper into the robustness application for the TSPTW and VRPTW.

3.1. Attended home delivery problem

The development of the past 20 years allows people to purchase food and other groceries online and utilize the convenience of a home delivery service. This new online grocery service, also called e-grocery, has had a large impact on the food supply chains (Ogawara et al., 2003; Agatz et al., 2008). Due to its convenience, it has grown substantially over the past years into a dominant distribution channel of business-to-consumer e-commerce (Campbell and Savelsbergh, 2006). However, a big challenge faced in the e-grocery is the perishability and the need for proper storage for most food and drink products. As opposed to the standard package delivery, food and drinks need to be stored properly at the customer. This limits the ability to deliver products at the neighbors, or hide it behind objects at the customers property when they are not at home. Therefore, deliveries require the attendance of the customer at the moment of delivery (Hsu et al., 2007). In order to prevent absence of the customer at the time of delivery, retailers are allowing their customers to choose their own preferred time slot. Due to this attendance, the problem addressed is in the literature is called the attended home delivery problem (AHDP) (Ehmke and Campbell, 2014). Research has shown that retailers with a home delivery service face a difficult logistical challenge (Fernie et al., 2010). The use of time windows comes with the challenge of actually meeting the agreed time. Resulting traffic or other delays form a hard logistical challenge for retail companies. These insecurities can lead to high delivery costs, waste of waiting time and lower customer satisfaction. At the stage of transportation, the AHDP has mainly been addressed as a vehicle routing problem, or a vehicle routing problem with time windows if a delivery time window is imposed (Ehmke and Mattfeld, 2012; Hsu et al., 2007). When only solving single trips without the need of looking at multiple vehicles, the problem is addressed as a large set of Traveling salesman problems with time windows (TSPTW).

3.2. Vehicle routing problem

The vehicle routing problem (VRP) is a problem within transportation and deals with a set of geographically spread delivery and/or pick-up points, restricted to a certain number of constraints (Laporte et al., (2013). The idea is to compute an optimal route between these customer nodes by minimising certain characteristics, like total distance, amount of costs or traveling time.

The VRP is an NP-hard problem in combinatorial optimization (Lenstra and Kan, 1981). That means that the problem cannot be solved within polynomial time. Therefore heuristics are used to find good, near-optimal solutions within a reasonable time. The origin of the problem is not exactly known within the literature. Stories of somewhat likewise problems date back to the 1930's, when Merrill Flood tried to obtain near-optimal solutions in reference to routing of school buses (Dantzig et al., 1954).

One of the most famous versions of the VRP is called the traveling salesman problem (TSP). Within this version there is only one depot, one vehicle with unlimited capacity and no restrictions present. A simple visualisation with six customer locations and a depot (in green) can be found in Figure 17. The increase in complexity for the TSP variation of the VRP can be seen from the following equation that shows the distinct tours for an Euclidean TSP: $\frac{(N-1)!}{2}$ where N is the number of customer locations to visit (including the begin/end point) (Vos. 2016). The simple version of the problem visualised in Figure 17 with six customers and the same start and end depot has a solution space of 360 unique routes. When expanding this to 13 customers as shown in Figure 18 this results in over 3 billion possible routes. This example shows the complexity of a VRP.

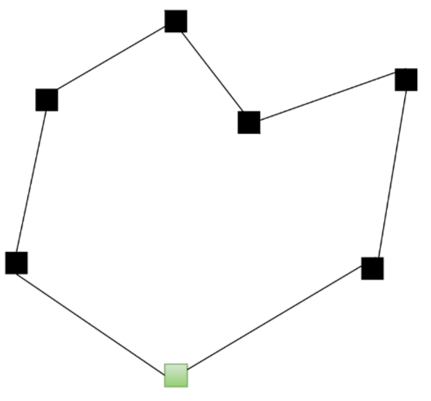


Figure 17: Simple version of TSP

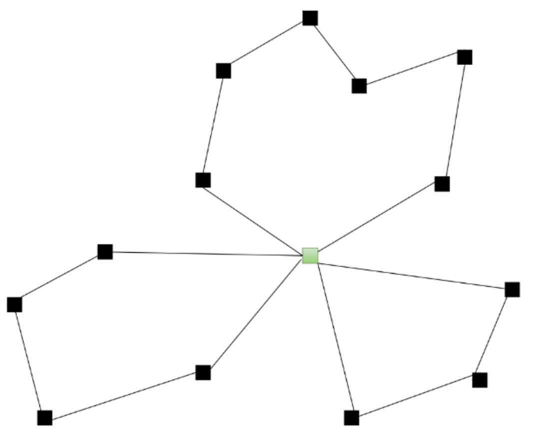


Figure 18: More complex version of TSP

Over the years many adaptations of the VRP emerged. Figure 19 provides an overview of the most common variants of the VRP and the relations between them as mentioned by Toth & Vigo (2002).

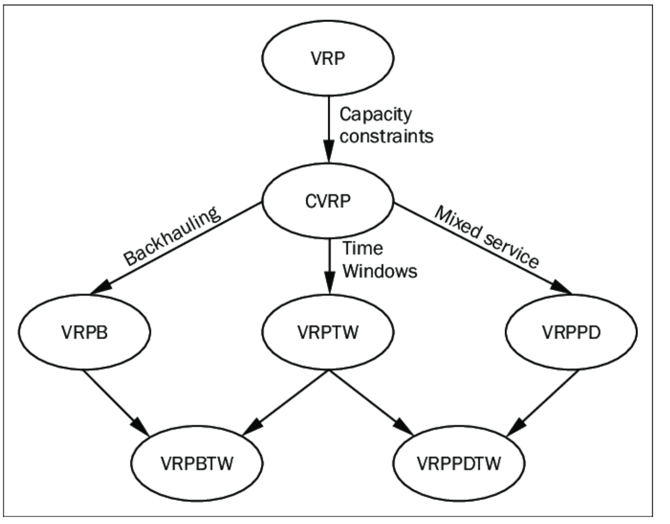


Figure 19: Variations on the VRP by Toth & Vigo (2013)

At the top of the overview we start extending the VRP with the use of capacity constraints (CVRP). This adaptation to the VRP constrain vehicles to only have a limited amount of capacity, which limits the amount of locations to visit (Sultana et al., 2021). After this, the problem can be extended in three different ways. First off the inclusion of backhauling (VRPB). Here the vehicle needs to collect goods after it completes all of its deliveries (Jacobs-Blecha and Goetschalckx, 1992). Next up, the inclusion of time windows (VRPTW) where customers can determine certain time slots in which the delivery needs to take place (Li et al. 2009). Lastly we have the adaptation of mixed service, where the problem will be modelled with the inclusion of pick-ups (VRPPD). This includes next to the normal deliveries, customers with goods that need to be transported back to the depot (Yanik et al. 2014). The problem with backhauling and mixed service can also be extended with the use of time windows which will create two extra adaptations of the VRP (VRPBTW and VRPPDTW). Other VRP adaptations relevant for this research topic are the Dynamic Vehicle Routing Problem (DVRP) (Hildebrandt et al. 2021) and the Vehicle Re-scheduling Problem (VRSP) (Mirchandani and Borenstein, 2007). Many other variations exist, however they can in most cases be categorized in one or multiple of the variants mentioned above.

3.3. Vehicle Routing Problem with Time Windows

When we are looking at the home delivery sector from an e-grocery perspective, we are dealing with the attended home delivery service (AHDS) as customers must be present for the delivery due to the perishability of goods. Within the AHDS, it is therefore common for the company to offer the customer a choice of a certain delivery time slot. These time slots both help the customer and the company to experience a better service level and to avoid delivery failures. However, this makes the process of delivering much more challenging as the plan and route all depend on the time slots of customers. This will make the Attended home delivery problem (AHDP) part of the VRPTW as we need to deliver goods to customers within certain time windows. Confirmed by the literature, we see that the AHDP is indeed mainly addressed as a VRPTW (Campbell and Savelsbergh, 2005; Agatz et al., 2008; Ehmke and Mattfeld, 2012; Hsu et al., 2007; Pan et al., 2017).

3.3.1. Application to our problem

To narrow down our scope of the project, we determined that we are going to adapt the current plan of the HDC by only looking at rearranging the stop sequence within a single trip (intra-trip mutation). This means the VRPTW problem we address, will basically become a large set of single TSPTW problems that need to be solved. This creates a simpler VRP, as we do not have to take vehicles and their capacity into consideration. Within problems with time windows, there are two kinds of classes, namely the use of hard and soft time windows. The difference between the two is that for hard time windows customer stops may not be delivered outside the time window, which often results in the use of extra vehicles. Whilst with soft time windows one can exceed the time window limit at certain penalty costs.

Hsu et al. (2007), give insight in how soft time windows can be used in the AHDP. They revised the common used relationship between arrival time, time windows and penalty costs shown in Figure 20. Here, [r,s] represents the given time window of the customer. The difference with regards to hard time windows, is that there is only a certain amount of time the delivery can be early or late in order to preserve customer satisfaction. After that period, shown as [R,S], a large penalty called M is introduced to avoid this occurrence. The revised version, shown in Figure 21, assumed the operator would rather wait and serve on time, since the increased cost is negligible. Consequently, R is approximated to r, similar to hard time-window constraints. The probability, that the order can be delivered within time when it is planned after the time window has exceeded, depends on the amount of time it arrives after the end of the time window. This probability decreases, at an increasing rate, as the time of delivery nears a certain value S.

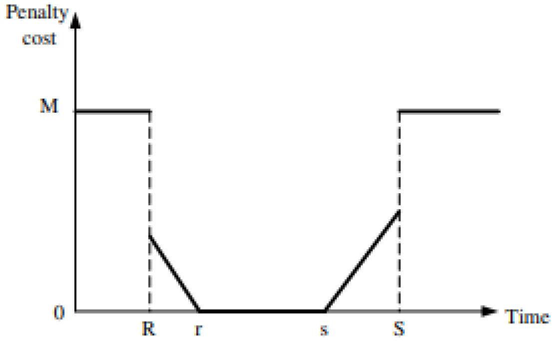


Figure 20: The relationship between arrival time, time-windows and penalty cost.

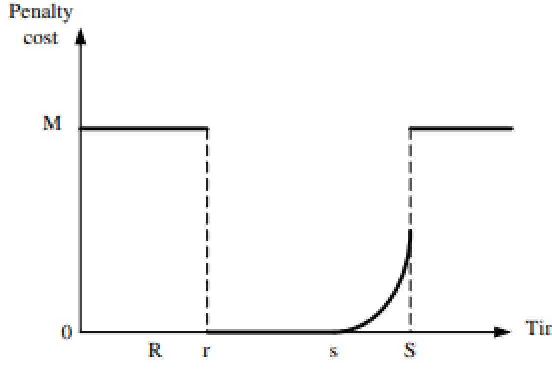


Figure 21: The revised relationship between arrival time, time-windows and penalty cost.

When we look at the current plans, we see that a small part of the stops is planned outside the time window. This suggests that there is a possibility of delivering late and that we are able to use soft time windows. Next to that, we do not have information about the fleet of the HDC and additional costs. So, we want to optimise the plan without adding any additional vehicles or personnel. Therefore, for our algorithm, we will not be looking into hard time windows and adding extra vehicles, but further focus on the use of a certain penalty cost for exceeding the given time window. Next to that, we also try to focus on the hybrid version as shown in Figure 21.

3.4. Algorithms for solving a VRP

Solving the TSPTW is a time-consuming process as the problem is stated as NP-hard (Jiang et al., 2020). In order to solve the TSPTW other approaches have been used, such as heuristics. In order to create a good heuristic, Cordeau et al. (2002) determined four main attributes that are crucial:

I. Speed - The amount of time it takes for the heuristic to find a solution for the given instance.
II. Accuracy - The deviation of the heuristic's solution with regards to the optimal value.
III. Simplicity - The heuristic should be understandable in order to be implemented.
IV. Flexibility - The amount of flexibility the heuristic has to be implemented for different problem sizes and environments.

In our case we need to deliver an improved plan within the time of receiving the plan and executing it. Therefore we need to keep track on the speed of our heuristic. Accuracy also plays an important role, as the heuristic's solution need to create plans that are significantly more robust. Simplicity and flexibility are variables that need to be taken into account but are less important for our heuristic.

The TSPTW is solvable by using different techniques like exact methods, classic heuristics and metaheuristics. To help to get some insights into some techniques of solving a VRP and their application, we are presenting some examples in the following subsections.

3.4.1. Exact methods

Cordeau et al. (2006) created an overview of some exact algorithms that can be used for solving a VRP. Most common methods are Branch-and-cut, lagrangean relaxation and column generation. Branch-and-cut uses the branch and bound algorithm while using cutting planes to tighten the linear programming relaxations. The lagrangean relaxation creates an approximate solution while simplifying the original problem by penalizing inequality constraints using a lagrange multiplier. Furthermore, the column generation method only generates variables that look promising to improve the objective function. The method is based on the fact that only a small group of variables need to be considered in order to solve the optimization problem.

Because the TSPTW is a problem considered as an NP-hard combinatorial optimization problem it is difficult to solve. Therefore solving the TSPTW with exact methods as mentioned above is very time consuming and only work for small problem instances. A single trip, within the plan of the HDC, consists next to the start and end on a hub, of dozens different customer stops. Ignoring the possibility of time window constraints, the solution space of a single trip is extremely large. As we have a restricted amount of time and near optimal solutions will also suffice, we will focus more on heuristic approaches for our problem instance.

3.4.2. Classic heuristics

Classic heuristics are used in order to quickly obtain feasible near optimal solutions for different instances. Within classic heuristics there are two different types defined, constructive and improvement heuristics. The goal for constructive heuristics is to find a solution from scratch, while improvement heuristics start with an initial solution which will be further improved (Laporte et al., 2000).

For our problem instance we already received an initial solution for the TSPTW by the customer. Thus using a constructive heuristic and start over from scratch again would not make much sense. Also this will take more time which we could use to improve the initial solution. Therefore we will not go in depth into constructive heuristics but only give a short introduction.

3.4.3. Constructive heuristics

A constructive heuristic starts with an empty solution and extends the solution piece by piece until a complete solution has been formed. Some of the most used constructive heuristic for routing is the Clarke and Wright savings heuristic (Clarke and Wright, 1964). Here the heuristic starts with creating for each customer a separate route from the depot to the customer and back to the depot.

Subsequently customers will be inserted into existing routes by determining the corresponding savings. Here certain restrictions like time windows or capacity constraints are taken into account. Other frequently used heuristics are the sweep algorithm from Gillett and Miller (Gillet and Miller, 1974) and the nearest-neighbor heuristic as defined in the paper of Solomon (Solomon, 1987).

3.4.4. Improvement heuristics

As mentioned earlier, an improvement heuristic starts with an initial solution and tries to improve this solution by modifying routes. This is very relevant for our research as we are improving a plan that already exists. This plan from the customer can therefore be seen as our initial solution from which we will start improving.

Frequently used improvement heuristics are local and global search methods. These methods create neighboring solutions from the existing solution by using a move-generation mechanism that will change one or more attributes of the current solution. The method accepts or rejects the created neighboring solution based on certain improvement conditions of the newly found solution (Bräysy and Gendreau, 2005a). Many different types of neighborhood heuristics exists. El-Sherbeny (2010) created a list of the most used neighborhood heuristics. Two commonly used heuristics for the TSPTW are the 2-opt operator and the 1-shift operator. Different literature with TSPTW problems experiment with using these two operators (Küçükoğlu et al., 2019; Voccia et al., 2013; Ma et al., 2019; Carlton and Barnes, 1996).

3.4.5. Metaheuristics

Metaheuristics are often used techniques that are powerful for a large number of problems. Nasser (2010) performed research about some metaheuristic methods for the VRP. He gives us the following definition of a metaheuristic: "*A Metatheuristic refers to an iterative master strategy that guides and modifies the operations of subordinate heuristics by combining intelligently different concepts for exploring and exploiting* *the search space."* A metaheuristic looks for an optimal solution within the search space in an iterative way. Some examples of commonly used metaheuristics are genetic algorithms, ant colony optimization, simulated annealing and tabu search. The success of these methods is due to their capacity ‘‘to solve in practice’’ some hard combinatorial problems. As these methods are promising to be used for our problem, we perform some extra research and give some insight into four different metaheuristics.

Genetic Algorithms (GAs)

GAs are algorithmic models which are population-based and inspired by the theory of genetic evolution of Darwin (Holland 1992). The solution of an optimisation problem using the GA methodology involves a stochastic search of the solution space using strings of integers, known as chromosomes, which represent the parameters being optimised. GAs have been widely used by researchers to solve single and multi-objective optimisation problems (Jalili et al., 2021).

Ant colony optimization (ACO)

ACO is a metaheuristic that is inspired by the pheromone trail laying and following behaviour of some ant species (Dorigo et al., 2006). When searching for food, ants initially explore the area surrounding their nest in a random manner. After they found food, they will bring back a portion of it and lay down a chemical pheromone trail depending on the quantity and quality of the food. The pheromone trails enables them to find shortest paths between their nest and food sources. Ant colony optimization exploits a similar mechanism for solving optimization problems. The method aims to concentrate the search in regions of the search space containing high quality solutions (Dorigo and Blum, 2005).

Simulated annealing (SA)

SA-based algorithms are popular local search metaheuristics used to solve single and multi-objective optimization problems, where a desired global minimum or maximum is hidden among many local minima or maxima (Suman and Kumar, 2006). The key feature of simulated annealing is that it provides a means to escape local optima during its neighbourhood search, by allowing so called “hill-climbing” moves in the hope of finding a global optimum. These moves allow the objective function to degrade based on a temperature value that determines if the worse solution is accepted. The temperature value decreases along the way which decreases the chance of accepting a worse solution. The annealing method with a temperature, denotes a physical process where a solid is first heated up and then slowly cooled so that when eventually its structure is "frozen," this happens at a minimum energy configuration (Van Laarhoven and Aarts, 1987).

Tabu Search (TS)

TS is an iterative procedure for building an extended neighbourhood with particular emphasis on avoiding being caught in a local optimum (Barbarosoglu and Ozgur, 1999). Tabu search applies restrictions to guide the search to diverse regions. These restrictions are in relation to memory structures which remembers different worse solutions for a certain amount of iterations. Tabu search has obtained optimal and near optimal solutions to a wide variety of classical and practical problems in applications ranging from scheduling to telecommunications and from character recognition to neural networks (Hertz et al., 1995).

From literature we find that specifically tabu search is a very powerful technique in solving different VRP variants (Liu et al., 2013). Computational results on Solomon's benchmarks proved that the proposed tabu search is comparable in terms of solution quality to the best performing metaheuristic Algorithm for the VRPTW published heuristics (Ochelska-Mierzejewska, 2020). The tabu search algorithm in essence, repeatedly searches the solution space by going from one solution to another in its neighborhood. Therefore it is able to handle the different time windows emerging within the VRPTW quite well.

3.5. Robustness within planning

Robustness is used in various contexts and can be interpreted in many different ways. In order to improve the robustness for our home-delivery plan, we need to know the definition of robustness. Next to that, we need to perform some research about the use of robustness within planning, whereafter we can further discuss the implementation of it within the TSPTW.

As mentioned in the research of Billaut et al. (2008), "*scheduling problems can be found in many domains. Almost every sector is concerned by scheduling problems in the broad sense:*

- *Industrial production systems: problems may need to be solved simultaneously in machine scheduling and vehicle dispatching (automated guided systems, robotic cells, hoist scheduling problems), in workshop layout problems or supply chain management problems.*
- *Computer systems: for example, to make full use of the processing power provided by parallel machines or when scheduling tasks with resource constraints in real-time environments.*
- *Administrative systems: appointment scheduling in health care sector, general resource assignment, timetabling, etc.*
- *Transportation systems: vehicle routing problems, traveling salesman problems, etc."*

According to the dictionary, scheduling is the art of planning activities so that one can achieve your goals and priorities in the available time. Nevertheless, often uncertainties happen along the way which can have impact on the execution of the plan. Therefore, one would like a plan to be 'robust'. In the literature, the robustness of a plan is often defined as its ability to perform well under dynamic and uncertain operational environments (Billau et al., 2008; Dooley and Mahmoodi, 1992). Dooley and Mahmoodi also created a graphical representation of how the concept of robustness applies to scheduling problems. The explanation of Figure 22 is as follows: "*The 'non-robust' system, as noted by the straight line, directly transmits all the dispersion in processing times to an equal amount of dispersion in throughput times. The robust system, as noted by the non-linear curve, has the ability to compensate for the dispersion in processing times and yields both a lower average throughput time and throughput times with less dispersion.*"

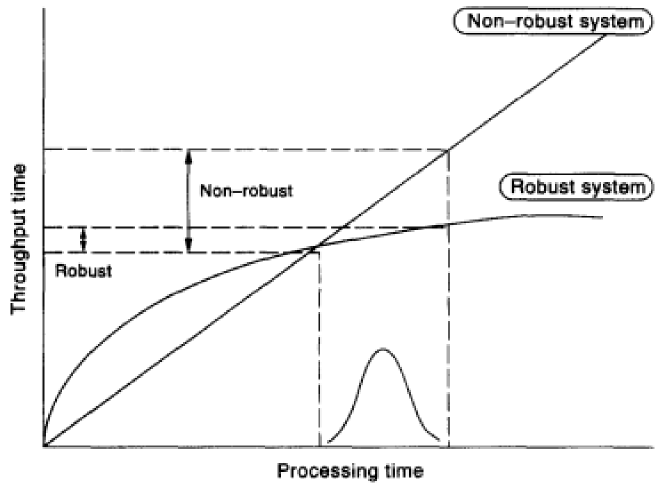


Figure 22: Robustness within scheduling problems by Dooley and Mahmoodi (1992)

3.5.1. Robustness within VRPTW

For the VRPTW, uncertainties have large consequences for the on-time deliveries. Therefore robust plans are required in order to increase the probability of delivering within the given time windows. Uncertainties have been made a hot research topic by the duo of Razmjooy and Ramezani in the last years (Razmjooy and Ramezani, 2018; 2019a; 2019b; 2021). To the best of our knowledge, there have not been many studies that did research on robustness improvement for the VRPTW within the attended home delivery sector concerning e-groceries where time windows are (near) fixed. The main uncertainty within this case is the influence of traffic delays on the travel time. In 2021, two papers were published within the literature that focus on robustness for the VRPTW (Tan et al., 2021; Duan et al., 2021) and one for the TSPTW (Bartolini et al., 2021). For the VRPTW both papers used the Robust multi-objective evolutionary algorithm (R-MOEA) from He et al. (2019) in order to simultaneously optimize the total travel distance and the number of vehicles required for transport. However both papers create routes from scratch and do not take algorithm speed into consideration. This makes their use for our problem instance with regards to the TSPTW instead of the VRPTW quite difficult. The paper about TSPTW from Bartolini et al. gives us a good head start on how to increase robustness for a TSPTW. However the paper mainly focusses on reducing costs under uncertain travel and service times rather than increasing robustness for a trip.

We find another application of robustness optimisation within the aviation sector. This sector also deals with strict time windows in which aircrafts need to depart in order to prevent possible delay. Marla et al. (2018) performed this robustness optimization within aircraft routing. Because we want to optimising the robustness by changing the customer sequence, this can be very applicable for our problem. They define robust plans as those plans that are less likely to be inoperable as planned due to disruptions occurring during the course of operations. Due to irregular and unpredictable traffic we also need to create plans based on the same criteria. They use a robustness KPI of minimising the probability of a flight being delayed. By using chance-constrained programming (CCP) they are re-arranging the slack in the plan to place it where it is most needed. This method is very promising for the problem we face. In order to optimise robustness for the home delivery problem, we want to maximise the probability of delivering within the given time window, or in this case, minimising the probability of delivering late.

3.5.2. Implementation of robustness

As mentioned in Section 3.5.1, we see multiple possibilities in order to implement robustness. Various papers implement robustness by using a multi-objective algorithm which simultaneously will optimise robustness by reducing travel uncertainties and a second variable like travel distance or certain costs. We see this in for example the paper of Tan et al. (2021) and Duan et al. (2021). Another method found in the literature is the paper of Zhang et. al (2021), who use a data-driven model in order to optimise the robustness using their own created Service Fulfillment Risk Index (SRI). Multiple of these self-created methods based on travel time uncertainties arise within literature (Nasri et al., 2020; Hu et al., 2018). We also found some applications where robustness is used as a constraint within the algorithm, an example is the paper from Wang et al. (2019), where one can set their own preferences of robustness.

3.6. Conclusion

The VRPTW problem is a widely studied subject over the past 25 years. We found numerous methods that have been created in order to solve the problem. We find applications of exact methods, classic heuristics, constructive heuristics, improvement heuristics and metaheuristics. We determined that the capabilities of metaheuristics apply very well to our problem instance as they proved their ability to solve some hard combinatorial problems. Especially the tabu search method is a widely used method for VRPs with time windows. Its ability to deal with different time windows will help find solutions within reasonable time.

When looking at robustness improvement, we can conclude from our literature that robustness within the VRPTW for the home delivery sector is a fresh subject within existing research. The robustness of a plan is often defined as its ability to perform well under dynamic and uncertain operational environments. Due to the extensive growth of the home delivery sector we see a couple of papers on this subject emerging over the past years. These papers are creating good robust VRPTW solutions, however, the used architectures do not meet the characteristics of accuracy, speed, simplicity, and flexibility that we need for our specific case. As Simacan receives daily plans only a couple of hours before execution, time is limited and starting from scratch will not create feasible solutions in time. Our study therefore will focus on creating a new method in order to adapt existing plans by using an improvement heuristic. The method used in the paper of Bartolini et al. (2021) and the method within the aviation sector (Marla et al. 2018), gives us a good indication of how this can be achieved. By re-arranging slack to where it is most needed we can improve our robustness and maximise the probability of on-time deliveries.

4. Problem description and solution approach

In this section the solution problem and approach is explained. We start by given a problem description in Section 4.1. Then, in Section 4.2, we give an introduction in the robustness increase. Section 4.3 provides insight into the approach of the solution and the steps we followed. Section 4.4 describes the model assumptions we took to be able to create a working model. Section 4.5 starts by introducing the used adapted model of the VRPTW. The objective robustness function of the VRPTW is explained in Section 4.6. Next, Section 4.7 explains the tabu search algorithm we used to optimise the plan and lastly, Section 4.8 explains how we solved the routing and travel time calculation for the new routes.

4.1. Problem definition

In this section, the notation of all variables, parameters and sets is introduced. We follow the order process from a customer's perspective as described in Figure 5. This consist of mainly three different steps: (i) arrival of the customer, (ii) location identification, (iii) time slot selection. On a daily bases customers will arrive at the online website of the HDC in order to select their groceries and place an order to be delivered at their home on a certain day. A customer i is part of the set of customers N, i.e., $i \in N$. As mentioned in Section 2.1, The arrival of customers can happen at any time in the horizon $[0,T]$. After this time T, the plan for the next day will be created by the planning instance of the retailer. Customers will order a certain amount of groceries which comes with a certain size and weight indicated by volume, q_i. Each customer also has its own delivery service duration, i.e., the time it takes for the deliverer to hand over the delivery to the customer after it has arrived at the destination. This service duration is denoted by S_i. The service time we will use is acquired from the original plan received by the retailer.

After the groceries have been selected, the customer must log in, or if ordering for the first time enter the desired delivery address. The address is used in order to assign a customer to a delivery hub. A hub h is part of the set of hubs H, i.e., $h \in H$. From each hub all assigned customer orders will be planned into different trips starting and ending in that specific hub. Furthermore, the customer has the possibility to select a time slot in which the delivery will take place. The time slots vary in lengths and are spread throughout the day. After time T when all orders have been received, a plan will be created accordingly by resolving the resulting VRPTW.

4.2. Robustness increase

The plans created for the HDC mentioned in the previous section are focussed on minimising the travel distance within the delivery of the time window. An effect of this is that many customers are delivered near the end of their time window as we showed in Section 2.1. As a result, there is a high probability that a stop will be delivered late if there is any delay during the trip. Looking at the traffic intensity curve from Figure 2, traffic intensity has developed rapidly over the past 20 years. This has had a large impact for the HDC on transportation due to the increase of delays. Due to this and the extensive growth of e-grocery customers in recent years during Covid-19 times, the HDC has had a change of direction and wants to focus more towards on-time delivery. The HDC has asked Simacan to give insights on how to realise and monitor these changes.

At present, the plans for a trip of the HDC are not being executed as they are made by the planning instance. Delivery drivers receive a trip's schedule and focus on delivering every customer on the list within their time window. The drivers will therefore intuitively execute the plan depending on what they think is best. Here, we see the differences emerging between the very experienced drivers and all other drivers. Experienced drivers oversee the entire trip and sometimes deliberately choose to pass a customer to meet other customers' time windows, even if the customer is on the route. Unlike the less experienced drivers they create their own robust planning through using their experience. Unfortunately the amount of experienced drivers are small and the bulk of the drivers do not have this insight, which results in more lateness. Furthermore, as all delivery drivers are executing plans in their own way it is hard to get insight into execution if everyone just does what they think is right.

To increase the chance of on-time delivery, plans need to become more robust in order to withstand occurring delays. To measure the robustness improvement of the plans, we use the probability of a late delivery as a KPI. In order to determine the probability, we can use data from Simacan of historical trips from the HDC. As illustrated in Section 2.4 there are some significant differences between delivery hubs located in different regions over the Netherlands. To be able to overcome these differences, we need to analyse every hub and approach the trips of each hub differently through creating their own robustness function.

4.3. Solution Approach

As mentioned in Section 1.4, the objective of this research is to increase the robustness of a home delivery plan. From the literature, we determined that the problem addressed can be qualified as a VRPTW problem. Due to the fact that we already have an existing plan, we further address the problem as a TSPTW, where we adapt the plan of a single trip. This means we do not have to take into account the amount of vehicles to use and their capacity allocation. In order to increase robustness, our approach is to use an algorithm which increases robustness based on historical data from Simacan. From the literature we found out that tabu search is a promising algorithm to use for the VRPTW. Therefore, we decided to use this algorithm in order to optimise robustness. The goal of the algorithm is to find better robust solutions by rearranging the stop sequence of the current plan. The algorithm consists of 4 phases. Figure 23 shows the algorithm outline with the use of a flow chart of the four phases. We start in phase 1 by creating a new, or use the original plan as a feasible starting solution. Within phase 2, we need to prepare the data. We create new route and distance matrices to speed up the algorithm during execution and propose specific robustness functions for each individual hub of the HDC, as we have seen in Section 2.4 that they are quite different from each other. After we finish all previous steps, we can perform the tabu search method within phase 3. From the literature, we determined that a 2-opt neighbourhood search should work well as operator for the tabu search algorithm.

The tabu search method creates (near) optimal solutions based on the robustness function, with the use of realistic travel times from Simacan's Traveltime API. Finally, we can visualise the new trip sequence to analyse the improvements in phase 4.

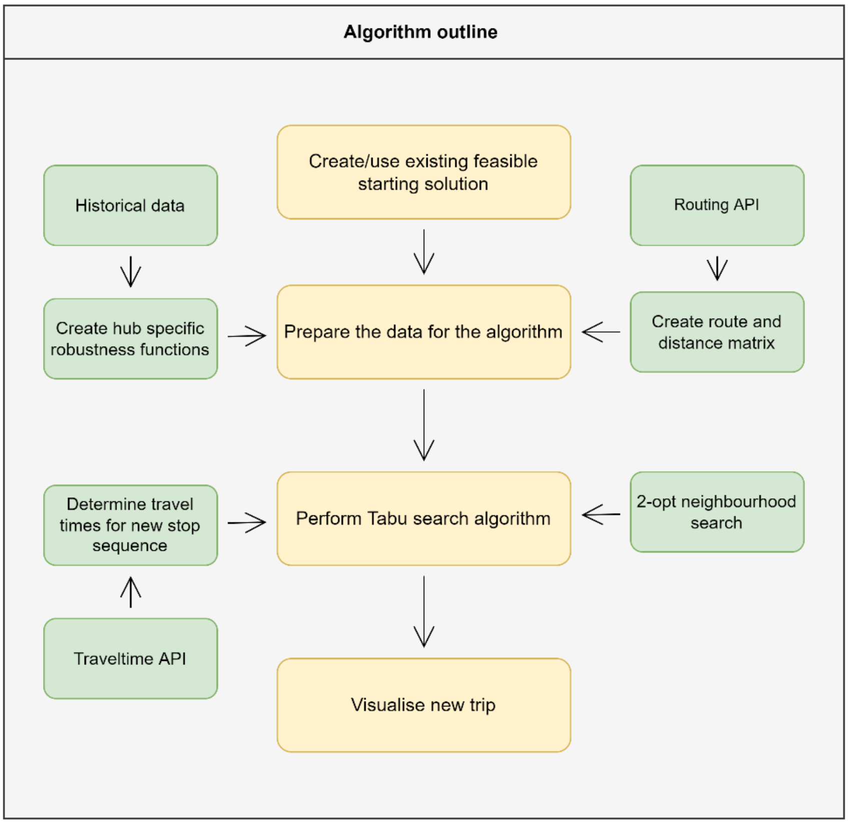


Figure 23: Flow chart of the proposed algorithm outline for solving the robustness of the TSPTW

4.4. Model assumptions

The objective of the algorithm is to increase the robustness of a trip by changing the stop sequence of customers while still using the same resources. In order to simplify the modelling of the problem and scope it down, several assumptions are made:

1. In order to reduce the complexity of the problem, preserve algorithm speed and due to a lack of data, we determined that we will only focus on optimising the sequence by looking at intra-trip and not inter-trip modifications. This means that exchanging stops between different trips will not be part of this research.
2. During a trip drivers need to take breaks. These breaks are implemented in the current plan. However, in practice, breaks are not held at the suggested time within the plan which makes it very hard to predict the arrival times at customer locations. Therefore we will not be implementing breaks within our model and exclude them from the original planning.
3. When a driver arrives at a customer it has to look for a suitable parking spot and unload the order from the truck. After that, he/she has to travel to the door and hand over the order to the customer before he/she is able to leave for his next customer stop. This time at the customer is called the

service time. Within our research, we assume that the existing fixed service time used within the original plan is accurate and therefore can be used within the optimisation model.

4. In most cases, when optimising the plan on robustness, other routes emerge. These changes can make a trip longer in terms of distance the truck has to travel. The amount of extra distance travelled by the vehicle through rearranging the stop sequence is not primarily optimised within the model. However, we deliver insights in the increase in kilometers and will perform some experiments on it.
5. As mentioned before, within our optimisation routes can become longer and take more time to complete. On a daily basis a vehicle will execute multiple trips. We assume that any extra travel time caused by the rearranging of stops, does not affect any subsequent trips, in terms of truck and personnel availability.
6. The planning of customer stops is based on time windows selected by the customer itself. Within our optimisation model we assume, when a stop is planned and delivered within the customers time window, that the customer is always present within that time window to receive the delivery.
7. As mentioned in Section 1.5, we will use the Routing and Traveltme API from Simacan to determine routes and travel times between stops. We assume these routes and travel times are optimal for a home delivery vehicle and return accurate predictions of travel times.
8. Within our optimisation model we rearrange stops within a single trip. We assume that this rearranging does not result in any capacity or storage issues within the delivery truck and does not involve any extra loading or unloading time.
9. We determined for each hub of the HDC a different formula for our robustness function. This lateness probability function of a single hub is based on a combination of multiple years of probability values from historical hub data. We assume this data will be accurate for future trips starting from a hub.

4.5. Vehicle routing problem

In order to use the basic VRPTW model for our problem we need to make some adjustments. As mentioned in the literature study we are minimising the probability of a late delivery. The main change lies within the objective function, this needs to be changed to a robustness function as we want to optimise the solution in terms of a probability. When implementing all changes on the standard VRPTW this results in a revised model as shown below. A further explanation of the objective function based on probabilities, can be found in the next section. The model minimises the robustness value while putting constraints on departing from the hub, visiting all customers from the set and vistiting them within their given time window. The model consists of the following parameters and variables.

Parameters

i	The customer to depart from $i \in N$
j	The customer to travel to $j \in N$
s_i	Service time at customer i

t_{ij} Travel time from customer i to customer j $(i, j) \in A$

<u>Variables</u>

x_{ij} Is equal to 1 if arc (i, j) is used, and 0 otherwise

w_i Specifying the start of service at customer i

<u>Model:</u>

$$Min \sum_{(j)\forall A} Q_j(y_j) \tag{1}$$

Subject to:

$$\sum_{j\in\Delta+(i)} x_{ij} = 1, \qquad \forall i \in N \tag{2}$$

$$\sum_{j\in N} x_{0j} = 1 \tag{3}$$

$$\sum_{i\in\Delta-(j)} x_{ij} - \sum_{i\in\Delta+(j)} x_{ij} = 0, \qquad \forall j \in N \tag{4}$$

$$\sum_{i\in\Delta-(n+1)} x_{i,n+1} = 1 \tag{5}$$

$$x_{ij}(w_i + s_i + t_{ij} - w_j) \leq 0, \qquad \forall (i,j) \in A \tag{6}$$

$$a_i\left(\sum_{j\in\Delta+(i)} x_{ij}\right) \leq w_i \leq b_i\left(\sum_{j\in\Delta+(i)} x_{ij}\right), \qquad \forall i \in N \tag{7}$$

$$x_{ij} \geq 0, \qquad \forall (i,j) \in A \tag{8}$$

$$x_{ij}\ binary, \qquad \forall (i,j) \in A \tag{9}$$

The objective function (1) of this formulation expresses the robustness function. Constraints (2) restrict the assignment of each customer to exactly one vehicle route. Next, constraints (3)-(5) characterize the flow on the path to be followed. Additionally, constraints (6) & (7) guarantee planning feasibility with respect to time window considerations. Finally, (8) and (9) impose all flow variables need to be 0 or positive and all flow variables are binary.

4.6. Robustness function

Within Section 2.4 and 2.5, we emphasized the significant differences in timelines between the hubs of the HDC. In order to create a solution that is realistic for every scenario, we cannot take a single robustness function for all replanned trips. We need to create different functions for each hub of the HDC as they differ too much. Therefore, we have to analyse all hubs separately to come up with a suitable fit for each one of them. The robustness function we created is based on the time window penalty function. In the following subsections we explain how this is created.

4.6.1. Probability function

The development of the traffic situation within the Netherlands is very unclear due to past years of Covid-19 and the rapid growth we had before that. De Ruiter (2021) emphasizes this uncertainty and mentioned that it will take a while for this to become clear. This makes it hard to predict how traffic will evolve in the coming years. In order to create these distributions for all the hubs of the HDC, we will focus on using historical data from multiple years. We created the distributions from four years of historical data, which includes two years without Covid-19 and two years with Covid-19. For every single hub, we want to create a separate distribution which will show the probability of a late delivery based on the amount of time planned before the end of the customers time window. Figure 24 gives an example of a distribution for a hub. The X-axis represents the amount of time (in minutes) the stop is planned before the end of the time window. The Y-axis gives the corresponding probability of a customer delivery being late if it would be planned that amount of minutes before the end of the time window. From this figure we can see that the lateness distribution of this hub follows an inverse exponential distribution. When the stop is planned more and more towards the end of the time window, the probability of a late delivery becomes very high. Therefore, to create more robust plans, we need to replan stops further from the end of their time window to decrease this probability.

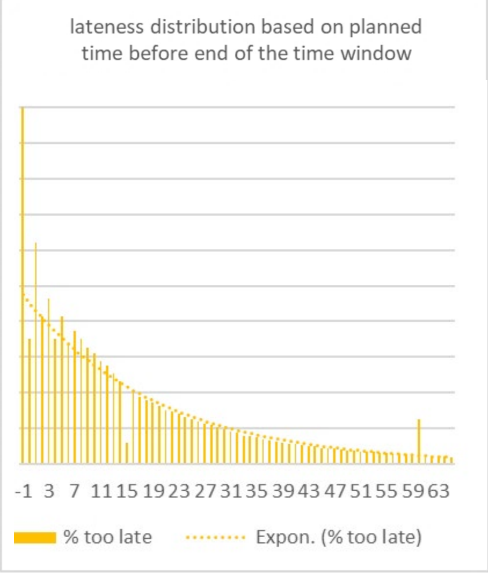


Figure 24: Distribution of lateness of a single hub based on historical data

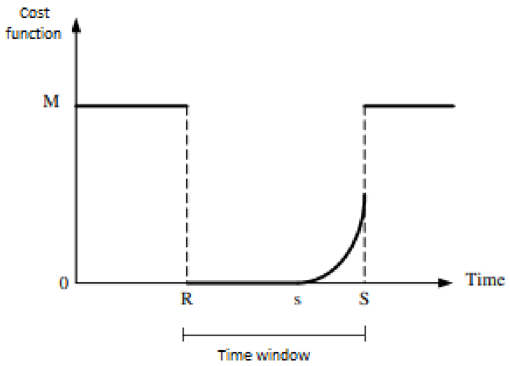
As we need to perform an increase on robustness for the entire plan, which consists of multiple delivery hubs, we need to come up with multiple functions in order to give each hub its own specific robustness function. As mentioned in Section 2.4, the HDC has multiple hubs. We looked at each hub individually in order to create the right probability function. From here we concluded that the distribution of all hubs follows an inverse exponential distribution with parameters α and λ. Hence the probability density function of a single hub (h) is given by;

$$f(x) = \alpha_h * e^{\left(-\frac{1}{\lambda_h}x\right)}$$

The probability functions of every hub with parameters α and λ can be found in Appendix A.

4.6.2. Robustness function

To improve robustness of home delivery plans we will make use of the penalty function based on the paper of Hsu et al. (2007) mentioned in Section 3.3. We adapted the function to be applicable for our robustness case. This function determines a robustness value based on the number of minutes a customer stop is planned before the end of the time window. The robustness function follows from the probability functions determined in the previous section based on historical data of each hub. Next to the probability function we also consider the customers time window. As can be seen from the lateness distribution from Figure 24, the chance of a late delivery will become very low when planning it further from the end of the time window. From the data we see that after a period of around 65 minutes, the probability will stabilise. This is due to the fact that after this time period, only very large delays have an impact on the timeliness of delivery. Traffic delays that result in this large amount of delay are very rare and do not occur very often within the home delivery sector, as most of the trips take place within a city. Most of the delays that have an impact after this time have nothing to do with traffic flow, but are related to accidents or truck breakdowns and malfunctions. The final robustness function will therefore consist of four parts. Figure 25 gives a visual example of what this looks like with the corresponding formulas.



$$Q_j(y_j) = \begin{Bmatrix} M, & y_j < R_j \\ 0, & R_j \le y_j \le s_j \\ \alpha_h * e^{\left(-\frac{1}{\lambda_h}x\right)}, & s_j < y_j \le S_j \\ M, & y_j > S_j \end{Bmatrix}, \quad j = 1, \dots, n,$$

Figure 25: Visualisation of the Robustness function with corresponding formula

The robustness function is structured as follows: the time window of customer *j* is given by the start time *R* and end time *S*. Within the time window, there is a certain point *s* at which the chance of not delivering on time starts increasing based on the lateness probability function $f(x)$. As mentioned, this point starts for most hubs at around 65 minutes before the end of the time window. This lateness function exponentially increases and becomes larger when reaching the end of the time window. Because the main goal is to create a plan where all customers are planned within the time window, we prevented the algorithm to be able to plan a customer stop outside the time window. This was done by setting the robustness value for planning outside the time window to a so called big *M*, instead of continuing to use the lateness function. This robustness value *M*, is calculated so that the algorithm can never prefer to plan a single customer outside its time window. This *M* value has been used after the end of the time window, but also before the start of the time window. Alternative options for the big M value is later on experimented with.

From the beforementioned four parts, the robustness function has been created. The robustness function presented by $Q_j(y_j)$, is based on y_j being the number of minutes planned before the end of the time window at customer *j*.

4.6.3. Robustness value

The robustness function mentioned in the previous section, is used to determine the robustness value. The robustness value of an entire trip is calculated by the delivery time of each customer within the trip.

<u>Customer level</u>

The value on a customer level is based on when a customer is being delivered with regards to their time window. The function returns for a single customer stop in the trip, the probability of a late delivery based on historical data. Note that this function is not linear, but increases near the end of the time window. When the stop is planned to be delivered outside the time window, it returns a very high penalty value. This is done to prevent a preference for the algorithm to have 1 extreme late delivery at a single customer while all other customer deliveries within the trip are on time with very low robustness values. The robustness on the customer level is used to calculate the robustness on the trip level. The lower the value the better.

<u>Trip level</u>

The robustness value on a trip level is used within the optimisation algorithm. As the goal of our research is to have a robust plan, we need to have a low robustness value on a trip level. Therefore, we need to combine the robustness values of all customers within a single trip. To calculate the robustness value of an entire trip, we decided to add up all individual robustness values of the customer level to a total robustness value that represents the robustness of the trip. This means, the lower the outcome of the robustness value, the better robust solution we get for the plan of a trip. A negative effect is that we are not able to look back on a customer level when examining the robustness value of a trip. However, we do know that with a lower robustness value at the trip level, we have a more robust solution in the end.

4.7. Tabu Search

The vehicle routing problem that we address in this research is stated as a complex problem due to the fact that it is an NP-hard problem in combinatorial optimization (Lenstra and Kan, 1981). The solution space is very large which makes it hard to solve. The plans are given a few hours before the trip is to be conducted, which makes it even more difficult. Therefore, in order to get a solution within reasonable time and near optimal, the problem can be solved by using a heuristic. In the literature study we determined that Tabu search is a common metaheuristic for solving the VRPTW for the home delivery sector (Liu et al., 2013; Jiang et al., 2020; Voccia et al., 2013; Ma et al., 2019). It is a good method to handle the different time windows which restrict customer stops to be delivered within a certain period of time. Locations that cannot be visited due to their time window become tabu for a certain period which helps the algorithm find better solutions quicker and search within a wider solution space.

To get started with the tabu search method we need to have an initial solution. Simacan will receive a plan from the HDC which hold all trips with needed information. This plan file holds a solution that can be used as the initial solution for the algorithm. Within Section 5.4 we will experiment with other starting scenarios to determine which starting scenario will perform the best for our problem.

Our tabu search method will be implemented with one operator. Based on the literature study we determined that the 2-opt operator provides good results for the VRPTW. The 2-opt operator swaps two stops within a trip. In Figure 26, we can find a small example of how the 2-opt operator works. With a certain selection method, point C and E are getting selected. The 2-opt operator will swap these two stops within the trip, which results in a new sequence. Hereafter the new sequence will be evaluated.

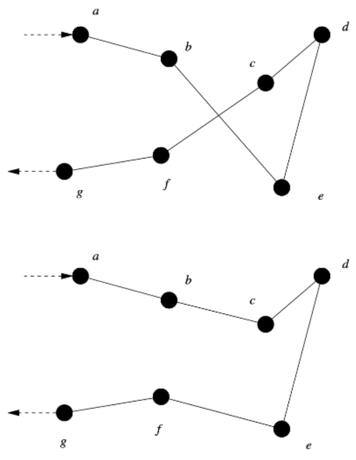


Figure 26: 2-opt swap example

From the tabu search method, new routing sequences are created from which we need to determine the robustness value. We will evaluate these new sequences by using our constructed robustness function. From here we know the new robustness value, which we use to optimise the route to have the lowest value of robustness. The robustness function takes only the probability of on time delivery in consideration and not the extra distance driven.

In order to execute the tabu search algorithm, we need to determine a couple of parameters. The algorithm is executed in a loop that needs a certain stopping criterion in order to determine when it has optimised enough. For our problem we created two different stopping criteria. First, a maximum amount of iterations and second, a threshold for non-consecutive improvements. When one of the two stopping criteria has been met, the algorithm has finished optimising. Another parameter that needs to be defined is the length of the tabu list. The tabu list holds a certain amount of swaps and will forget the first added tabu item when a new tabu item needs to be added while the list is full.

In Figure 27, the pseudo-code is provided for the used tabu search algorithm. In line 1 we start with constructing an initial solution S. In our case a starting solution already exists, namely the current plan. This is used as the initial solutions for the algorithm. Line 2 and 3 initialises certain parameters to zero, set the tenure value, and set the best solution as the starting solution. Line 4 creates a while loop that loops until at least one of two criteria are met. These criteria are: the actual consecutive non increasing iterations are larger than the maximum amount of consecutive non increasing iterations and, the amount of performed iterations is larger than the maximum amount of iterations. Line 6 performs the best possible swap and updates the current solution, which result in a new solution S’. If S’ is a feasible solution in line 6, we start comparing the robustness value of the best solution to the new solution in line 7. Line 8 and 9 are resetting the consecutive non increasing iterations, and update the best solutions and iterations after we find a better solution. When we find an equal robustness solution in line 10, we check the difference in distance for both the solutions in line 11. Next, if we find a better distance we reset the consecutive non increasing iterations, and update the best solutions and iterations in line 12 and 13. When we do not find a better distance, we only update the consecutive non increasing iterations and iterations in line 15 and 16. When we do not find an equal or better robustness solution, we also only update the consecutive non increasing iterations and iterations in line 18 and 19. After that in line 20, we update the tabu list. This means we add the performed swap to the tabu list and, if the list count has reached the tenure value, delete the first swap on the list in line 21. Next, we update the current solution with the new solution in line 22 and repeat the process until one of the stopping criteria is met and we return the best solution in line 24.

Algorithm TS for P-VRPTW

1.	Construct an initial solution S
2.	$C_{cni} = 0, C_{it} = 0, S_{best} = S$
3.	Set Tabu Tenure value
4.	**While** $C_{cni} \leq \in_{cni}$ or $C_{it} \leq \in_{it}$
5.	Select and apply the best/random non-tabu swap to S to obtain S'
6.	**If** S' is a feasible solution
7.	**If** $f(S') < f(S_{best})$
8.	$S_{best} = S', C_{cni} = 0$
9.	$C_{it} = C_{it} + 1$
10.	**Elsif** $f(S') = f(S_{best})$
11.	**If** $\text{Total distance } f(S') < \text{Total distance } f(S_{best})$
12.	$S_{best} = S', C_{cni} = 0$
13.	$C_{it} = C_{it} + 1$
14.	**Else**
15.	$C_{cni} = C_{cni} + 1$
16.	$C_{it} = C_{it} + 1$
17.	**Else**
18.	$C_{cni} = C_{cni} + 1$
19.	$C_{it} = C_{it} + 1$
20.	**End if**
21.	Update tabu list
22.	$S = S'$
23.	**End while**
24.	**Return** S_{best}

Figure 27: Pseudocode of the tabu search algorithm

As mentioned in the previous section, a couple of parameters like for example, $\in_{cni}$ (threshold of consecutive non improving iterations) and $\in_{it}$ (maximum amount iterations) need to be determined beforehand. As shown in the algorithm outline in Figure 23, before the algorithm can be executed for a single trip, some extra steps are involved to pre-process data and improve the speed of the algorithm.

- Import the plan file of the HDC with all stop data
- Pre-process stop data to right formats where it can be used
- Implement TravelTime- and routing API client

- Import probability functions of all hubs to be used within the robustness function
- Create routing and distance matrix where all optimal routes between customers will be determined and their corresponding distance

4.8. Routing and travel time

The rescheduling algorithm with the tabu search heuristic takes the original plan as an input. However, to be able to use the robustness function we need new arrival times. These are calculated with the use extra data including new optimal routes between the customer stops and the corresponding travel time on the time of day. This section will explain how this is done using two different APIs from Simacan.

4.8.1. Routing API

When customer stops get switched around, new optimal routes have to be created. To create these new optimal routes between two customers we are using the Routing API of Simacan. This service guarantees to return, with a certain vehicle profile, the best route on the map between two locations that can be used to calculate accurate travel times. It provides endpoints for requesting routes between latitude and longitude locations, returning the optimal route between these points when one is available. The corresponding route can be returned in different types, including coordinates, downstream and inward path, OpenLR, Google Encoded Polyline and WKT. All of these types can be used to visualise the route in different programs. When requesting a route, the Routing API supports three different profiles, Quickest, Truck and Home Delivery. The Quickest profile returns an optimal route for regular cars. The Truck profile is optimized for larger trucks and tries to avoid narrow and bendy roads, as well as roads where large trucks are not allowed due to their dimensions or weight. The Home Delivery profile is similar to Quickest, but has extra restrictions for dimensions. The Home Delivery profile include some extra restrictions like tunnels that home delivery vans cannot fit through.

4.8.2. Travel time calculation

To calculate an accurate travel time of a given route from the Routing API, we are using the TravelTime API of Simacan. This service is a collaboration between TomTom and Simacan and is used to predict accurate travel times based on historical or current traffic data. The content covers highways, urban and rural arterials, and secondary roads in more than 40 countries throughout Europe, North America and beyond. The TravelTime API consists of two different profiles that can be used in order to determine accurate travel times, namely “current” and “speed profile”. The profile to be chosen, depends on the moment in time the accurate travel time is needed. The current profile, also called HD Flow, aggregates real time speed data from millions of anonymous consumer GPS devices, providing true average speeds on individual road segments to determine the real travel time at the current moment in time. A visual representation is shown in Figure 28.

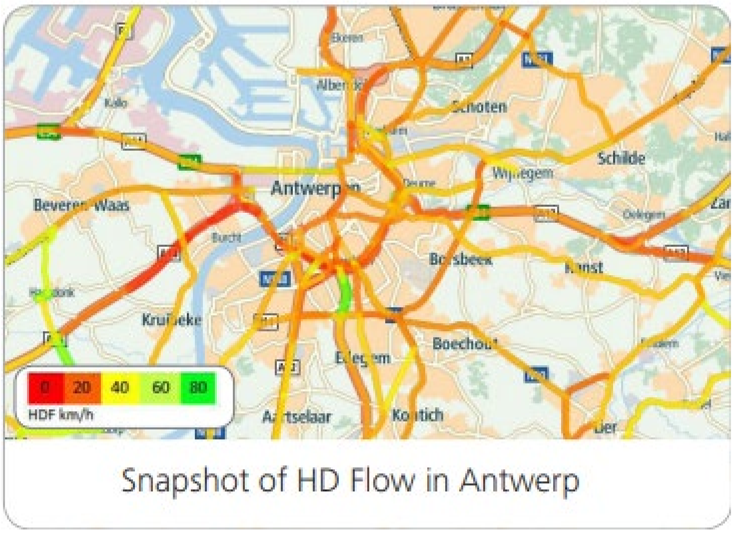


Figure 28: Visual representation of HD Flow profile of the TravelTime API

Next to the current profile, we have the speed profile. The speed profile captures traffic patterns for every five minute interval of a road section for each day of the week. This way of calculating the average travel time is based on two years of historical data. When using this profile, the travel time is being calculated by listing the average travel time from A to B. This is done by calculating for each specific road section the expected travel time at the specific time of the day the section will be passed. An example of a calculated travel time between A and B over different road sections of the speed profile compared to the traditional estimation is shown in Figure 29 and Figure 30. Figure 29 is a basic freeflow travel time calculation when one does not specify the time it will be driven. The different road sections from Figure 30 show diffent travel times for three different roads during the a certain time within morning peak. This could also be seen as a current profile when it was based on live congestion data. The average over small time periods from this is used as the speed profile where the expected travel times are based on historical data.

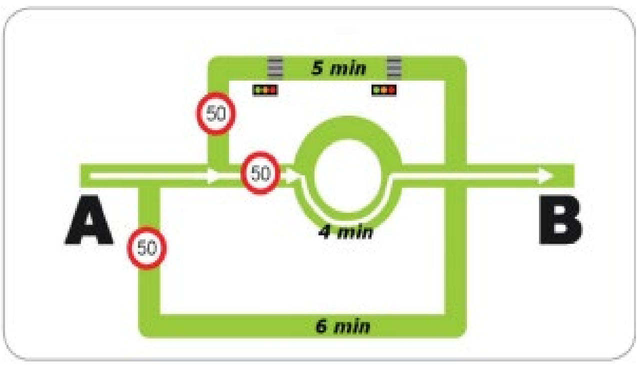


Figure 29: Traditional travel time estimation from A to B. (one value for the whole day)

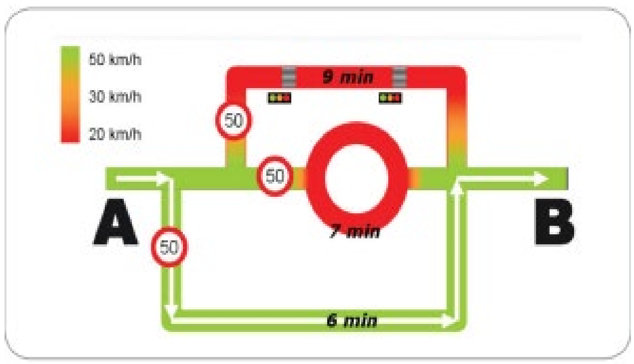


Figure 30: Example of average travel times from A to B during a time period in morning peak with the use of speed profiles

4.8.3. Implementation

In order to solve the given problem, we will need to use both the Traveltime API and the Routing API. Our problem takes place within the home delivery sector. Therefore, it makes sense to use the home delivery profile when sending routing requests to the Routing API. Also, in order to visualise the newly created routes we will extract the route as an OpenLR. With the OpenLR routing type, we can later on visualise the routes within the software of Quantum Gis. For calculating the corresponding travel time, we will use speed profiles. As our route is planned for the future, we do not know the exact traffic situation at that time of execution. Therefore we will make use of speed profiles which will predict the travel time based on historical data on each road section at each moment in time.

4.9. Conclusion

In this chapter we explained our proposed solution approach. From the literature we determined that a tabu search algorithm with a 2-opt operator is a good method for solving the VRPTW. In order to improve robustness for the HDC, we use the method of Hsu et al. (2007). We started by creating different lateness probability functions for each hub, which we used within the optimality function. This function consists of a probability function specifically determined for each hub between a certain amount of time, a period that there is no penalty and two periods where we have a certain big penalty costs 'M'. We then used this function as the objective function for the tabu search method and showed a pseudo-code of the overall algorithm with corresponding parameters. As during rescheduling with the tabu search algorithm we need to create new routes and travel times, we explained how these new routes will be created and exact travel times are calculated with the use of two APIs from Simacan. Finally we gave insight into some implementation assumptions we needed to make to be able to solve our problem.

5. Experimental design

In this chapter, we explain the design of the experiments. First in Section 5.1, we elaborate on the data set used for the experiments. Next, we briefly introduce the different experiments that we will perform in Section 5.2. Whereafter Section 5.3 will give insight on how the different starting solutions for an experiment are created. Finally, Section 5.4 finishes this chapter with an explanation about the multi-objective function used to combine distance and robustness.

5.1. Data set

In this section we will elaborate on the data set used for the different experiments mentioned further on. The data set is part of an existing plan from the HDC. Table 1 shows some of the main characteristics of the experimental data set.

Table 1: Overview of the characteristics of the experimental data set

Characteristic	Data set
Amount of trips	118
Min-max amount of customer stops within a single trip	10 – 33 stops
Min-max size of time window	1 – 5 hours
Operating time of trips	10:50 – 22:50
Day of the week	Monday
Max distance between stops	71 km
Number of covering Hubs	6

The characteristics of the data set above show the diversity of this experimental data set. We used this set to test the algorithm on both large and small variants of trips from a basic plan from the HDC. The data set consists of 118 different trips with a large variation in the amount of stops, time window sizes, total distance of trips and operating hours. The trips are chosen from 6 different hubs which are spread over the country in both urban and rural places. This way, we try to experiment with the algorithm to make sure it will function properly for (almost) all different scenarios that occur within the operation of the HDC.

5.2. Experimental setup

We will provide a detailed computational evaluation of the tabu search algorithm based on a set of TSPTW problems derived from existing trips from the HDC. We assess the effectiveness of the algorithm with respect to the different parameters and penalty functions. Subsequently we will try to obtain insights into the impact of distance on the solution and the resulting trade-off between robustness increase and distance.

All algorithms and API's mentioned are implemented within Python 3 language and implemented in PyCharm 64-bit version 2021.2.3. The computational experiments were run on a computer with an Intel core i7-9750H processor at 2.6 GHz and 16.0 GB of random-access memory. The experiments consist of five parts: we start with numerical experiments in order to optimise the different parameters. Secondly, we experiment with a composite penalty function, followed by an experiment on the impact of optimising on distance instead of robustness. Next, we try and use different starting solutions in order to evaluate the algorithms potential of escaping local optima. Lastly, we will perform some experiments with a multi-objective function, where we will combine distance and robustness within the objective function, in order to be able to set certain weights to both of the objectives.

Experiment 1: parameter tuning. The aim of the first experiment will be to optimise parameters for the created robust VRPTW model. We will try different tabu list lengths and stopping criteria in order to optimise speed and the robustness increase of the algorithm. We will experiment with large and small problem instances to ensure optimal settings for both instances. Due to the APIs that the model needs to call, we expect the model to be quite slow for large amount of iterations. However, we expect it to find better solutions in combination with larger tabu lists as it can better escape local optima. The best parameter outcomes will then be used in the following experiments.

Experiment 2: composite penalty function. The second experiment will focus on having a composite penalty function for delivering outside the time windows. This composite penalty function will be elaborated in Section 5.3. We need to adapt the penalty for being early on the stop to a lower value of that being late. In this way, the algorithm prefers being early rather than being late. When a driver would arrive early it can take a break or wait for a certain amount of time, whilst being late you do not have that kind of 'luxury' as you need to deliver as soon as possible. We implement and test a stepwise penalty function for being early on a customer stop, to make a difference between being one minute early in comparison to for example 10 minutes.

Experiment 3: optimising on distance. For the third experiment, we will adapt the algorithm to not optimise on robustness but to minimise the amount of travel distance for a single trip. This experiment will provide insights in how the optimisation on distance will affect the solution in terms of robustness. As we got the information that the plan of the HDC is primarily based on distance, we do not expect to find much improvement. However it will be quite interesting how an improvement on distance will affect the outcome of the robustness value. As we do only have to calculate travel times for the final route and already have a distance matrix for all stops, we expect this algorithm to be quite fast.

Experiment 4: alternative starting solution. Within the fourth experiment, we will create a method that uses different starting solutions for the algorithm. The current starting solution where we use the existing plan of the HDC is already optimised by their planning software. This optimisation can possibly lead to local optima when trying to improve robustness with our algorithm approach. The use of different feasible starting solutions helps the algorithm escape these local optima and find better solutions. The largest disadvantage of this method is that only finding a feasible starting solutions is already quite a challenge.

Experiment 5: multi-objective function. In this last experiment we aim to use the robustness algorithm in combination with distance reduction. We will implement a multi-objective function where we try to optimise on distance as well as on robustness at the same time. In this way our algorithm can be used while giving certain weights to the different objectives in order to optimise the plan. This can be used subsequently by Simacan to use the algorithm with weights specifically shaped for a customer. We will experiment with different weights in order to gain insights on how this will impact the outcomes.

5.3. Composite penalty function

As mentioned in experiment 2, we want to create a composite penalty function. We find multiple applications of a composite penalty function. For example within the paper of Halvorsen-Weare and Fagerholt (2010). They use a composite penalty function where the penalty outside the time window follows a stepwise pattern (Figure 31). Another application of different composite penalty functions we find in the paper of Slotboom (2019). As we only want to create a one-sided composite penalty function we adapted one of the penalty functions mentioned in the paper, which can be seen in Figure 32. For the experiment we determined to use this type of exponential penalty function to see how this affects our results. In order to penalise using the exponential function, we choose to use the following exponential function:

$$Exponential\ penalty = X * 1.5^{minutes\ before\ TW}$$

The function consists of 2 parts, a parameter X and a constant value of 1.5 to the power of the number of minutes the stop is scheduled before the time window. We decided this constant to be 1.5. This means every minute earlier has a penalty value that is 1.5 times more than its predecessor. Parameter X is a very difficult parameter to quantify. It determines the amount of acceptable minutes early on a customer stop without given large penalty values. Determining the right value for this is a business decision and is highly dependable on multiple aspects. Being early is a good thing when the customer is home, but results in unwanted costs if the driver has to wait. For the upcoming experiments, we will use a value of 0.001 for parameter X as this ensures a reasonable penalty until around 10 minutes before the time window.

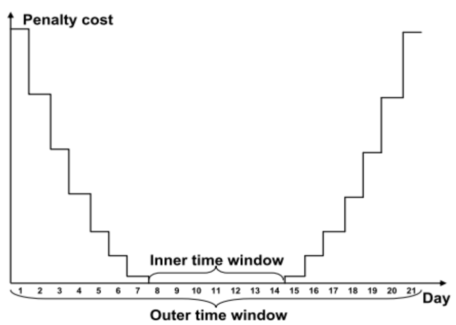


Figure 31: Stepwise penalty function by Halvorsen-Weare and Fagerholt (2010)

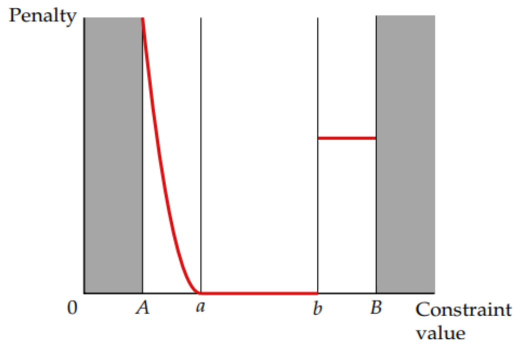


Figure 32: One-sided exponential penalty function based on paper of Slotboom (2019)

5.4. Alternative starting solution

In our proposed solution approach to the problem, we used the standard plan as it is delivered by the HDC. However as mentioned in experiment 3, we will also look at the effects of different starting solutions. Although, even building a feasible solution to the TSPTW is considered a NP-Hard problem according to Salvesbergh (1985). Therefore, in order to create new feasible solutions in reasonable time, we need to find a method that is able to do that. From the literature we found a constructive variable neighborhood (VNS) procedure created by da Silva and Urrutia (2010). The procedure is able to quickly find random feasible initial solutions in a fast way. In order to use the method we made some minor changes to adapt this to our specific problem. Figure 33 shows the adapted constructive procedure. The objective function that is used, is the sum of all positive differences between the time to reach a specific customer and the given time window, which is determined by the formula $\sum_{i=1}^{n} \max\left(0, \beta_i - b_i\right)$. We start the procedure by creating a solution through sorting the customers first on the earliest time window starting time and second on the time window size. Using this solution the algorithm works iteratively through line 6 to 12 until it finds a feasible solution (where the objective is equal to zero) or the max level is reached. In line 6 the solution is perturbed, by making level random 1-shift movements, to escape from the current local optimal solution. In line 7 a 1-shift local search procedure is applied to the perturbed solution, and in line 8 the solution obtained after the local search is set as the new best solution if it is better than the current best solution. At the end of each iteration, the level variable is increased by one if the current solution is not improved or reset to 1 otherwise.

Algorithm *VNS – Constructive phase*

1.	**Output:** X
2.	$Level = 1$
3.	$X = Initial\ solution\ (sorted\ first\ on\ time\ window\ start\ and\ secondly\ on\ size)$
4.	$BestX = X$
5.	**While** $X\ is\ infeasible\ and\ level < MaxLevel$
6.	$X^{'} = Perturbation(X, level)$
7.	$X^{'} = Local1Shift(X^{'})$
8.	**If** $X^{'} < BestX$
9.	$BestX = X'$
10.	$Level = 1$
11.	**Else**
12.	$Level \mathrel{+}= 1$
13.	**End while**
14.	**Return** $BestX$

Figure 33: VNS algorithm to create alternative starting solutions

5.5. Multi-objective function

As mentioned in experiment 4, we want to use an objective function that combines the distance and the robustness value in order to optimise on them simultaneously. From literature we find this application is called multi-objective optimization (MOO) (Marler and Arora, 2005). We find multiple methods of MOO like the weighted global criterion method, weighted sum method, lexicographic method and the goal programming method (Marler and Arora, 2004). Most methods use some kind of weighted value in order to optimise on multiple objectives. From the paper we find that the most common approach to MOO is the weighted sum method. Here one can assign weights to the different objectives in order to give importance to each one of the objectives. We decided to set the weights, such that the sum of the weights are equal to 1 and each weight has a positive value. When implementing this method for our case, the objective of the multi-objective function will be the following:

$$Min\, W_1 \sum_{(j)\forall A} Q_j(y_j) \;+\; W_2 \sum_{(i,j)\forall A} (X_{i,j} * Distance_{i,j})$$

Where $X_{i,j} = 1$ if the route between *i* and *j* will be used, and $X_{i,j} = 0$ otherwise. W_1 and W_2 are the weights given to each of the objectives.

When we want to use the above objective, we first need to perform some transformations on the values of distance and robustness. The distance and the robustness value cannot be simply added up. As the distance values are not from the same metric as the robustness values, the distance value will have significant more impact on the outcome than the robustness. Therefore, to prevent this overly influential variable, some transformation is needed in order to give robustness and distance the same impact on the solution. In order to do so, we will need to perform some kind of normalisation on the values to make them equally influential. From literature we found several methods. The most commonly used for our scenario in literature is called the 'upper-lower-bound approach' (Koski 1984, Koski and Silvennoinen 1987, Rao and Freiheit 1991, Yang et al. 1994). This method adapts the objective to a value between zero and one. This implies that the lowest value in the data will have the value 0 and the highest value in the data will have the value 1. All the other possible outcome values will be within this range between 0 and 1. To calculate the value we will use the following formula:

$$Z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)}$$

The normalised value Z_i is calculated from x_i, the outcome value for the robustness or distance, and the minimal and maximal value of x possible.

For the distance value, the minimum outcome will be 0 and the maximum will be calculated according to furthest distance. A common method used in the literature to find this furthest distance is called the farthest neighbor (Rahman and Rochan, 2016). This method determines the maximum distance by creating a new route by traveling to the farthest neighbor until all neighbors have been visited. In order to get the upper bound value we will use this method to find the longest distance for each trip. For the upper and lower bound of the robustness value, we will be using 0 as a minimum and for the maximum the worst case scenario where all stops will be delivered late. This value is different for every trip and will be calculated in advance.

5.6. Conclusion

In this chapter, we explained the setup for the different experiments. The data set we use consists of 118 trips with different characteristics, including the higher and lower extreme values in order to test the proposed algorithm. Further, we gave insight into the five experiments that we will perform, which help validate and test our proposed method. We will perform experiments for optimising the used parameters, test penalty adaptations on the objective function, test different starting plans in order to escape local optima, and finally, two experiments that will adapt the focus from optimising on robustness to solely distance and a combination of distance and robustness with a multi-objective function.

The alternative starting solutions will be created using a VNS procedure found from the literature. Here, we start by creating a solution through sorting the customers first on the earliest time window starting time and second on the time window size. Whereafter the VNS procedure creates new feasible solutions that we can use for our algorithm. For the multi-objective function we will make use of the weighted sum method, where weights will be assigned to each of the two objectives. The outcomes of the robustness and distance objective first need to be normalised as they do not have the same metrics. In order to do so we make use of the 'upper-lower-bound approach' where the objective values are normalised between 0 and 1 according to their minimum and maximum possible value. The maximum value of distance is consequently calculated by using the farthest neighbor.

6. Experimental results

This chapter will give insight into the outcomes of the experiments from Chapter 5. We start by showing the results from the parameter tuning in Section 6.1. In Section 6.2, we continue with the results from the composite penalty function. Section 6.3 points out the results of the distance optimisation, whereafter Section 6.4 displays the results of alternative starting solutions. Finally, Section 6.5 ends the chapter with a visualisation of experimental results from the multi-objective function.

6.1. Parameter tuning

In this section the result of experiment 1, the parameter tuning is presented. The tabu search algorithm that we created has different parameters that can be tuned to optimality. We performed 12 different experiments where we used different stopping criteria and tabu list lengths. In order to overcome random errors, we performed all experiments three times and took the average value. The average results of a single trip within the experiments can be found in Table 2 and Table 3. Within the tables, we show the number of iterations performed by the algorithm followed by the Consecutive non-increasing iterations (CNI) value. The next parts show results from running the algorithm with a certain tabu list length. The outcomes consist of the average running time in seconds and the average Δ increase in comparison to the original plan, for the length and robustness value of the trips. The robustness function adds up all lateness probability values of stops within a trip. Therefore, in order to have a better robustness value we aim to have the lowest possible robustness value. That means we are delivering less to the end of time windows for the customer stops within the trip. Hence, the robustness increase is shown as a negative percentage within the tables.

Table 2: Results from running parameter tuning experiments for tabu list lengths 50 and 100

		Tabu list length: 50			*Tabu list length: 100*		
			AVG. Δ increase(%)			*AVG. Δ increase(%)*	
Iterations	**CNI**	**Runtime (sec)**	**Trip Length**	**Robustness**	**Runtime (sec)**	**Trip Length**	**Robustness**
N = 500	250	100	9.82%	-25%	96	9.68%	-29%
N = 1000	250	159	8.09%	-28%	168	10.63%	-31%
N = 1500	250	321	9.62%	-29%	201	12.24%	-30%
N = 2000	250	474	12.72%	-32%	276	15.22%	-28%
Average:	250	264	10.06%	-28.5%	185	11.94%	-29.5%

Table 3: Results from running parameter tuning experiments for tabu list lengths 150 and 200

		Tabu list length: 150			*Tabu list length: 200*		
			AVG. Δ increase(%)			*AVG. Δ increase(%)*	
Iterations	**CNI**	**Runtime (sec)**	**Trip Length**	**Robustness**	**Runtime (sec)**	**Trip Length**	**Robustness**
N = 500	250	105	9.21%	-32%	101	12.09%	-28%
N = 1000	250	157	13.94%	-30%	159	8.65%	-35%
N = 1500	250	198	11.85%	-29%	192	9.81%	-35%
N = 2000	250	228	10.93%	-29%	281	11.51%	-33%
Average:	250	172	11.48%	-30%	183	10.52%	-32.75%

First of all, one would probably notice that the Consecutive non-increasing (CNI) value is the same for all experiments. The reason behind this is that during the experiments, we found out the algorithm will rarely reach a value more than 100. When stopping the algorithm at this lower value we experienced that it finds very poor solutions which negate the performance of the algorithm. We therefore decided to fix the CNI on 250, where it terminates the algorithm prematurely but gives it enough room to search the solution space.

When we observe the outcomes of the experiments we can draw several conclusions. The first observation is that it does not have a positive impact to the solution to run more than 1000 iterations. When looking at the outcomes of the 2000 iterations experiments, we see that in most cases we find a worse solution with regards to the trip length and its robustness. It also takes significantly more time for each trip to be solved. Furthermore, we can observe that if we increase the tabu list length we overall get better solutions in terms of robustness. This makes sense as the algorithm excludes more undesired swaps. When implementing a tabu length of over 200 we found that the algorithm would get stuck very often.

Overall we can conclude that the best possible parameter setup for the algorithm is to take a tabu list length of 200 and let the algorithm run for 1000 iterations. This decreases the robustness value on average with 35% with an increase in distance of 8.65%. Depending on the available time to run all trips through the algorithm, one could also use the setup with a tabu list length of 150 in combination with 500 iterations. This yields hardly any worse solutions at a significant lower running time per trip.

Figure 34 shows the robustness outcomes of the best setup for all the trips within the data set. The orange line represents the baseline measurement, which is the expected robustness value when the algorithm would not be able to find a better robustness value after running. The start robustness value refers to the robustness value of the original plan, and the end robustness to the value after running the tabu search algorithm. As can be observed from the figure there is quite some potential for improving the trips in terms of robustness. Most dots are below the baseline, which means the algorithm was able to find a better robustness value.

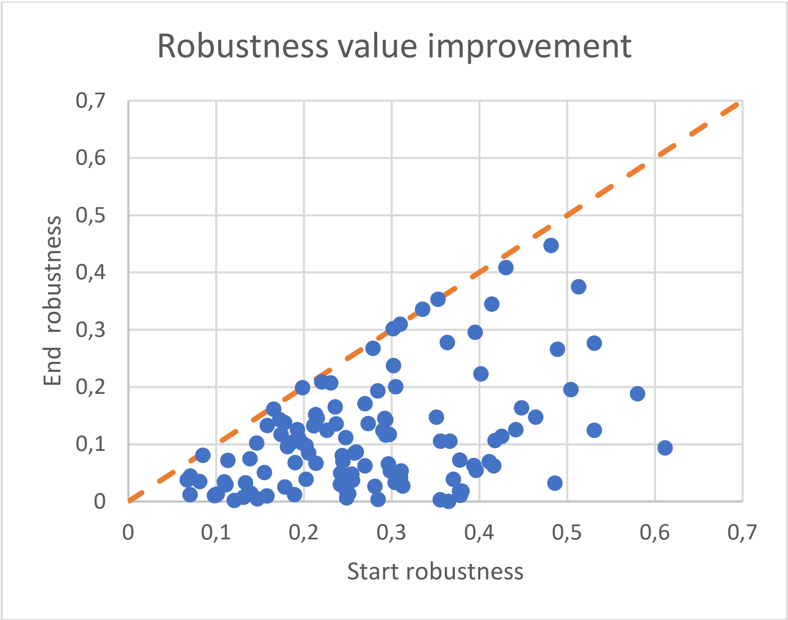


Figure 34: outcomes of the data set with a tabu list length of 200 in combination with 1000 iterations compared to the start robustness value

Since running the 118 trips of the data set with a tabu list length of 200 in combination with 1000 iterations will take over 5.25 hours, we decided to run the rest of the following experiments with the alternative setup with a tabu list length of 150 in combination with 500 iterations. This will save almost 2 hours per experiment, which we can use to perform more experiments to validate and optimise our method.

6.2. Results from the second experiment

In this section, we show the results of the second experiment, where we test an adaptation on our robustness function. We implemented a composite penalty function, that increases the penalty of being early on a stop by using an exponential function, while keeping a fixed costs for being late. Table 4 shows the results of the new penalty function in comparison with the standard function. Here we can conclude that the algorithm is only finding minor changes with the composite method. On average this leads to a better result for the length and robustness increase, but the median value becomes worse. In order to draw a conclusion from using this composite function, we looked further at the individual differences of the 118 trips within our data set. Showed in Table 5, we analysed the difference with regards to the combination of an improvement on the robustness and distance. For 56 out of 118 trips we determined that there was no significant difference of more than 0.5%, which comes down to about 47%. The remaining 62 trips in the data set, we divided under four different categories. A worse robustness but a better distance solution; a worse robustness and a worse distance solution; a better robustness and a better distance solution; a better robustness but a worse distance solution.

Overall, the remaining trips are almost equally divided over these categories. This makes it quite difficult to determine any significant improvement with using a composite penalty function over the standard. We therefore conclude, that having this different penalty function is something to look at within further research when breaks are also included within the tabu search. But for now, it does not add much to the solution of the current research.

Table 4: Results from using the one-sided composite penalty function on arriving before the time window

	AVG. Δ increase(%)		Median Δ increase(%)	
Function:	Length	Robustness	Length	Robustness
Standard	9.82%	-33%	8.15%	-39%
Composite	9.56%	-35%	8.82%	-38%

Table 5: Detailed overview for the influence of a composite penalty function on the trips within the data set

Combination robustness-distance solution		
Worse-Better	13	11%
Worse-Worse	18	15%
Better-Better	14	12%
Better-Worse	17	14%
No difference	56	47%

6.3. Results from the third experiment

In this section, we present the results from running the algorithm while optimising on the travel distance instead of robustness. In Section 5.2, we explained the standard planning of the HDC, as far as we know, is based on minimising the distance while still delivering within the customers time window. Running the algorithm on minimising distance, therefore gives us a good understanding if this is also really the case. After running the algorithm with the data set, we found out that on average the algorithm finishes improving a trip in about 45 seconds. Analysing the results of the new routes, we find on average an improvement of 3.2% on the distance of a trip. However, this results in 94% of the cases of at least 4 customer stops delivered far beyond their time window. Figure 35 and Figure 36 show the distance reduction and the increase of the robustness value when optimising on the travel distance. Within our data set, we found only a few minor exceptions that led to small improvements and not a huge increase in the robustness value. This experiment confirmed our expectation for the plan of the HDC. From within Simacan we have been told, they are optimising on reducing the travel distance while still delivering inside the time window. Whether this statement was really true is confirmed in the examples of Figure 35 and Figure 36. Figure 35 illustrates the outcome of the distance reduction. We see the total length of the trip from the start solution on the x-axis plotted against the percentage of the new solution of the trip after optimisation, with regards to the start solution. As mentioned before, we see minor optimisations are possible with an average reduction of 3.2%. Figure 36 illustrates the robustness value change when optimising on distance. The figure is based on the same principle as Figure 34, but with a logarithmic y-axis scale. From here we see that the decrease in trip distance lead to very high robustness values and thus unwanted deliveries outside their time window.

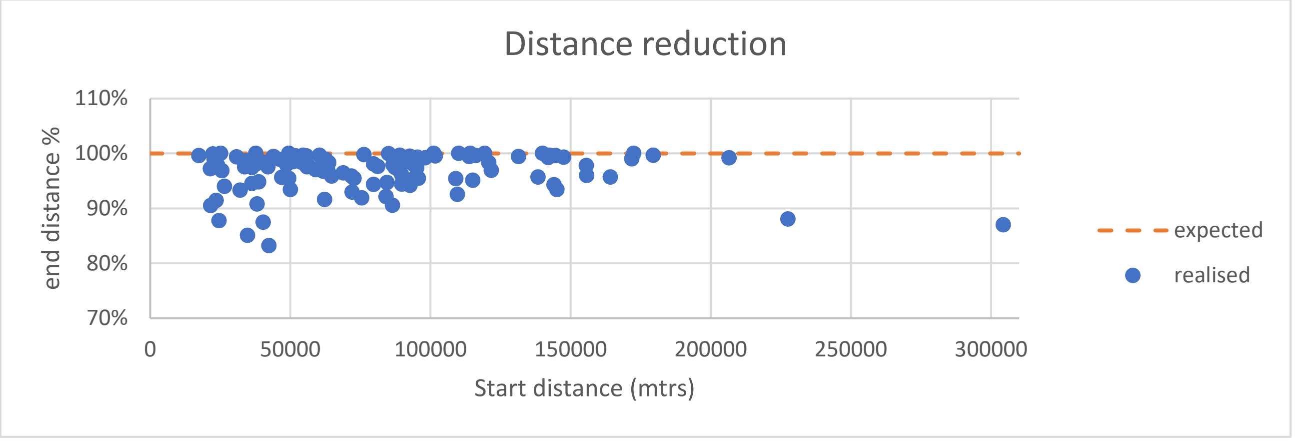


Figure 35: Result of distance reduction after running the algorithm to optimise on distance

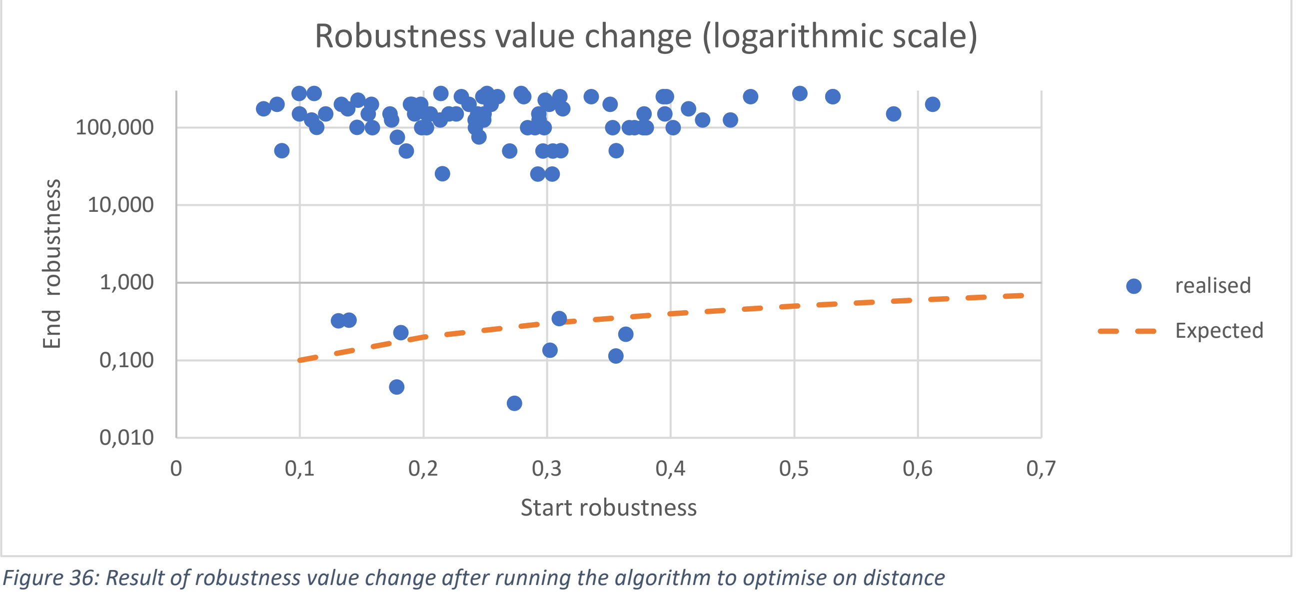


Figure 36: Result of robustness value change after running the algorithm to optimise on distance

6.4. Results from the fourth experiment

Table 6 gives an overview of the outcomes of using alternative starting solutions. As mentioned in Section 5.4, we created our own starting solutions for the plan of the HDC in order to look for improvements on the outcomes. When using alternative starting solutions, the algorithm has a better chance of escaping local optima within the solution space. From the table, we see that when giving the algorithm an alternative starting solution, we are able to find even better solutions in terms of robustness. Despite the original algorithm having anti cycling methods in order to explore more of the solution space, it still is not able to find the global optimum in cases.

When analysing the outcomes, we see that the average robustness value is lower and better with regards to using the original plan, but the median on the other hand lies much higher and is worse. We can explain this by the fact that around 10% of the trips cannot be solved. Here, the algorithm is not able to find good solutions within the given iterations and keeps getting stuck in a local optimum. The influence of the unresolved solutions has a large impact on the average value. When taking the median value, we can conclude that in most cases alternative starting solutions give better robustness improvements. However, it is difficult to say whether it is possible to use this option to solve the main problem, as using alternative starting solutions take considerably more time to solve each trip. Next to that, the algorithm also struggles to create feasible plans for some of the trips.

We chose not to carry out experiments with more than 1000 iterations, in order to try and get all trips resolved. After applying the algorithm with over 1000 iterations to a few trips, we saw that the total algorithm running time for a single trip increased significantly to more than 10 minutes per trip. Overall, this will not give us any results within reasonable amount of time and is therefore not a usable solution method.

Table 6: Outcome of alternative starting solutions

		AVG. Δ increase(%)		*Median Δ increase(%)*		
Iterations	**Runtime (s)**	**Length**	**Robustness**	**Length**	**Robustness**	**Unresolved**
N = 500	142	24.17%	-22%	20.35%	-50%	10%
N = 1000	328	21.97%	-23%	18.96%	-53%	9%

6.5. Results from the fifth experiment

Table 7 shows the results of three different setup variants for the multi-objective function experiment. As mentioned in Section 5.5, the goal of using a multi objective function is to optimise trips based on multiple criteria. In this case we determined that distance and robustness are two main important variables. By assigning a different weight values to each of these variables, the algorithm optimises the different trips according to the weighted objective. The table shows the average distance and robustness values from our data set before (on the original plan) and after executing the algorithm with the corresponding weighted objective values. The first observation is that, when using weights to optimise on multiple objectives, this ensures that the algorithm does not completely neglects the distance. This results in a much lower increase in trip length with regards to the solutions from Table 2 and 3, but still gives a very good increase in robustness for the trips. Overall, for everyday use this is probably the best method in order to increase the robustness.

Table 7: Results from the multi-objective function

Weights:		*AVG. before optimisation*			*AVG. after optimisation*		
Dist.	**Rob.**	**Trip length (m)**	**Robustness value**	**Weighted objective**	**Trip Length (m)**	**Robustness value**	**Weighted objective**
0.0	1.0	80324.70	0.276	0.276	90468.01	0.111	0.111
0.3	0.7	80324.70	0.276	0.322	86280.12	0.134	0.232
0.5	0.5	80324.70	0.276	0.353	83243.49	0.145	0.295
0.7	0.3	80324.70	0.276	0.384	81275.19	0.169	0.356
1.0	0.0	80324.70	0.276	0.215	77651.13	185.86	0.205

6.6. Conclusion

In this Section, we showed the results of several experiments we performed on our solution method. We started with improving the performance by optimising the parameters used within the algorithm. This showed us that there is great potential for improving robustness of trips within our data set. However, time plays an important role in the ability to run the algorithm. The remaining experiments are based on expanding and testing the algorithm. We performed experiments with a composite penalty function, where we found out that there is hardly any change when changing the standard objective function. Only small changes occurred within the robustness and distance objective, which can be related to randomness errors. The distance optimisation experiments showed us that the current plan from the HDC is indeed based on a distance optimisation. We were able to find minor improvements but at the costs of more late deliveries.

To test the performance of our algorithm, we did experiments with alternative starting solutions. This showed us that our algorithm is quite good at escaping local optima within restricted amount of time. The alternative starting solutions showed some better results, but overall needed too many iterations and showed on average less improvement. In our final experiment we changed the objective of the algorithm to a multi-objective function. In this way we showed insights in the ability to increase not only the robustness, but also do this at a minimal increase in travel distance of the trip. We showed results for three different weight distributions which gave promising insights for optimising robustness of trips without increasing the distance by much.

7. Conclusions

In this chapter, we summarise and discuss the research findings, draw conclusions and summarize both the practical and scientific contribution of this thesis in Section 7.1. After that, we give recommendations and discuss opportunities for future research on this subject in Section 7.2.

7.1. Conclusion

For this research, we explored the possibilities for improving the robustness of home delivery plans. We performed this case specifically for the plans of the HDC. In the last 10 years, the use of methods to buy groceries online and have them delivered to your home has experienced tremendous growth. Within the last two years of Covid-19, the amount of customers for the HDC grew rapidly. This creates a number of difficult problems within their plan with regards to on-time deliveries. The goal of this research was to develop a method in order to increase the robustness of a plan, where the robustness is measured by the probability of an on-time delivery. The model we built to solve the problem of the HDC is based on historical data gathered from the platform of Simacan. This data includes planned and realised arrival and departure times, traffic data and customer location and time window data. After analysing the data and performing research within the literature, we created an algorithm that uses a tabu search method which replans the trips based on their robustness value. To determine this value, we have developed distribution functions for each of the HDC’s existing hubs based on the probability of on-time delivery given a certain amount of time before the end of a customer’s time window.

To validate our method we used part of an existing plan of the HDC. This data set consists of 118 trips with different characteristics to be able to cover lots of different cases. With the use of this data set, we optimised the necessary parameters and ran all the experiments. To overcome random errors, we performed all experiments three times and took the average values as our result. First we have the parameter tuning. From the parameter tuning, we found out that our test data set shows lots of improvement potential with regards to the robustness. Using the algorithm, we are able to improve the robustness of a trip by almost 35% at the cost of a distance improvement of 8.65%. The algorithm will complete the optimisation with an average time of 2 minutes and 40 seconds per trip. After optimising the parameters, we performed four different experiments, where the original robustness model was tested and extended.

The goal of the first experiment was to extend the basic algorithm by implementing a composite penalty function within the objective function. We implemented a different penalty function for being early on a customer stop. We changed the fixed penalty to an exponential function which results in a small penalty for the first couple minutes early and increases exponentially. From the experimental results, we concluded that having a different penalty function does not have any significant effect on the solution for distance or robustness. We encountered some minor changes which we can relate to random errors.

To gain more insight into the current plans, the goal of the second experiment was to check if the plans from the experimental data set are indeed optimised on travel distance. The results of this experiment tell us that the plans of the HDC are indeed based on traveling the least amount of distance, while still trying to deliver within the time window. We were able to find some minor improvements on the current plans. However, these improvements led to multiple customer stops being planned to deliver far beyond their time window.

The goal of experiment 3 was to validate the algorithm in its capability to find good solutions. As we previously used the plan from the HDC directly as an input for our algorithm, we restrict it in exploring the solution space. Therefore, we wanted to see if the algorithm could perform better and find better solutions when giving it other starting solutions. The outcome of the experiments showed some interesting results. We can conclude that when we start optimising with alternative initial solutions, we are able to get significantly better robustness improvements. However, it is difficult to say whether it is possible to use this option in a real-life application. Improving the robustness of a trip by using alternative initial solutions take considerably more time. In addition, the algorithm also struggles with around 10% of the trips to create feasible solutions within the time limit of 1000 iterations. Performing more iterations in order to create feasible solutions for all trips result in run times that are out of proportion.

Finally we experimented with a new objective function for the tabu search method, that combines both distance and robustness into a single objective value. This combined objective will most likely be more applicable for the real-life case instead of using a single robustness objective. The HDC wants to improve the on-time delivery of their trips, but also want it at a minimal amount of cost increase. The results of the third experiment show that we are able to improve the robustness for some trips by 50%. However, this comes at the cost of increasing the distance by more than 20% which is very costly. The multi-objective function gives the user the opportunity to give weights to the optimisation of both the distance and robustness, in order to balance between optimising on robustness and distance. The experimental results from using this method with different weights, show us a very large improvement potential in robustness. We also concluded that optimising the robustness to the max, is not worth the extra distance increase of the trip. The 50/50 robustness distance weight optimisation gives us a good example of this viewpoint. Using these weights, we still get very good improvements while keeping the increase in distance to a minimum. Changing the weights to a 70/30 distribution results in only minor extra robustness improvements with regards to the 50/50 distribution, but has a much higher distance increase.

We only studied the effect of our solution approach on a single delivery day. Therefore, we made the assumption that the plans are independent of the day. Next to that the robustness function is also calculated independent from the day of the week. In practice, the reality of the situation may be questionable and will require further investigation. Also, we completely neglected the use of breaks within the current plans. Breaks can enable the delivery driver when he/she arrives before the start of the time window, to take a break before a stop in order to get a lower penalty on the robustness function. Next to that, leaving out breaks gives more slack within the plan which is in contrast with reality. We further assumed that any extra travel time caused by the rearranging of stops, does not affect any subsequent trips within the daily plan in terms of truck and personnel availability. Because of these limitations within the research, we cannot be certain about the actual effects of increasing the robustness. We only showed the potential effect for a best-case scenario where traffic flow continues its pattern based on historical data. With the plan of the HDC, we showed a real life application in terms of customer and traffic data and hub lateness distributions.

We can conclude that the robustness increase of home delivery plans based on historical traffic data has added value to the attended home delivery problem. When using the method on a home delivery plan of the HDC, a significant potential increase in robustness can be made. However, only to a certain extend they are useful within a real life application. From our experiments we determined that when improving on robustness alone, this does not outweigh the increase in travel distance. Using a multi-objective function shows far more potential for this case and should therefore be further investigated.

Summarizing, the contribution of this thesis to the scientific literature is twofold. Firstly, we developed a hybrid method by applying pre-trip robustness improvement with a tabu search method to the attended home delivery problem while using historical data. Secondly, we created a multi-objective application to optimise on multiple KPIs which makes it more applicable to the real life situation. The combination of these subjects is uncommon in existing literature which makes this project of potential added value to Simacan and literature. The contribution of this thesis to practice is that the results show the potential of increasing the robustness of trips based on historical data with a tabu search method. The implementation of realised historical data instead of simulations, shows the feasibility of the solution within a real life environment. This result contributes to the objective of the HDC and Simacan to give insight in the potential increase of on-time delivery of the plans from the HDC.

7.2. Recommendations and future research

Based on the conclusion of this research, in this section we suggest several recommendations to Simacan and the HDC. Furthermore, we give a couple of ideas for future ideas on the research subject.

With our experiments, we showed the potential of using the robustness increase method, and therefore advice to further investigate the possibilities to implement it on the current plans. We showed that the method used within this research has its limitations, but can be used to get an approximation of the potential improvements. It is recommended to use the multi-objective improvement option when using the algorithm. It gives good robustness improvements at the costs of some added distance which can be chosen by variable weights. When using this within a real-life environment, it is recommended to find a good balance between these weights. Next to that, with the use of Simacan's platform it is possible to create multiple simulations of the newly created plans. These simulations can help to confirm and determine the exact improvement potential by running the newly created plans on real-time traffic data. On the longer-term, we believe that the improvement based on historical data becomes outdated. Therefore the distribution functions of all hubs need to be updated frequently in order to retain the improvement potential. Finally, we have some recommendations regarding the algorithm we created. Our knowledge goes as far as using Python code to create a working algorithm. This coding language does work for this problem, but is quite slow and not optimised for these kinds of problems. The recommended multi-objective option takes around 160sec to optimise a single trip. With this speed, the algorithm is not able to finish an entire plan which is send around 10 hours in advance. The current plan is divided into two day-shifts and consists around 850 trips per shift, which results in a running time of almost 38 hours. Porting and optimising the method to other programming languages is therefore recommended to significantly increase the speed of the algorithm.

Further research can be done on a couple of subjects. First of all, we neglected to implement the use of breaks within the rescheduling of trips. The reason behind this is that Simacan is unable to detect breaks within their platform. In addition, the drivers do not take a break at the indicated times which makes it very difficult to implement a break. When Simacan is able to detect breaks, it is recommended to perform research on how to implement breaks within the rescheduling algorithm to further increase the feasibility of the revised plan. After this is implemented, the use of a composite penalty function to be able to take brakes when a customer stop is planned early, becomes also more interesting. The second recommendation for future research, is to try and expand the current method to perform robustness improvement for an entire plan using also inter trip mutations. Currently we only replan customer stops within a single trip and do not exchange customers stop between trips. When including inter trip mutations the improvement possibilities to increase robustness are most likely at higher computational complexity as well, opening opportunities for interesting trade-offs. Furthermore, this research is a stepping stone to perform further research into the subject of on-trip robustness optimisation. This research showed that robustness improvement has a high potential to optimise a plan. However using this method while the trip is “on-going”, is a whole new subject and comes with many more challenges. Using this optimisation in practice can lead to massive improvements and is very valuable for both the HDC and Simacan.

8. References

Agatz N, Campbell AM, Fleischmann M, Savelsbergh M (2008) Challenges and opportunities in attended home delivery. Golden B, Raghavan S, Wasil E, eds. The Vehicle Routing Problem: Latest Advances and New Challenges (Springer, Boston), 379–396.

Barbarosoglu G., D. Ozgur (1999), A tabu search algorithm for the vehicle routing problem, Computers & Operations Research 26, 255—270

Bartolini, E., Goeke, D., Schneider, M., & Ye, M. (2021). The robust traveling salesman problem with time windows under knapsack-constrained travel time uncertainty. Transportation Science, 55(2), 371-394. doi:10.1287/TRSC.2020.1011

Billaut, J.-C., Moukrim, A., & Sanlaville, E. (2008). Flexibility and robustness in scheduling. Wiley: New York.

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

Campbell A. and M. Savelsbergh (2005). Decision support for consumer direct grocery initiatives. Transportation Science, 39(3):313–327.

Campbell, A.M. and Savelsbergh, M. (2006), "Incentive schemes for attended home delivery services", Transportation Science, Vol. 40 No. 3, pp. 327-341.

Carlton, W.B.; Barnes, J.W. (1996). Solving the Traveling-Salesman Problem with Time Windows Using Tabu Search. IIE Trans. 1996, 28, 617–629.

CBS. (2020, November). Groei afgelegde kilometers in Nederland vlakt af. Opgehaald van CBS: https://www.cbs.nl/nl-nl/nieuws/2020/45/groei-afgelegde-kilometers-in-nederland-vlakt-af

CBS. (2021, Oktober). Personenauto's. Retrieved from CBS: https://www.cbs.nl/nl-nl/visualisaties/verkeer-en-vervoer/vervoermiddelen-en-infrastructuur/personenautos

Clarke G., and J.V. Wright. (1964). Scheduling of vehicles from a central depot to a number of delivery points. Operations Research , 12, 568-581.

Cordeau J. F., M. Gendreau, G. Laporte, J. Y. Potvin, and F. Semet, (2002). "A guide to vehicle routing heuristics," Journal of the Operational Research Society, vol. 53, no. 5, pp. 512–522.

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.

da Silva, R. F., & Urrutia, S. (2010). A general VNS heuristic for the traveling salesman problem with time windows. Discrete Optimization, 7(4), 203–211. doi:10.1016/j. disopt.2010.04.002.

Dantzig, G., Fulkerson, R., & Johnson, S. (1954). Solution of a Large-Scale Traveling-Salesman Problem. Journal of the Operations Research Society of America, 2(4), 393–410.

De Ruiter, J. & Intertraffic (2021). rethinking-traffic-management-after-covid-19. Opgehaald van Intertraffic: https://www.intertraffic.com/news/traffic-management/rethinking-traffic-management-after-covid-19/

Dooley, K. J., & Mahmoodi, F. (1992). Identification of robust scheduling heuristics: Application of taguchi methods in simulation studies. Computers & Industrial Engineering, 22, 359–368.

Dorigo M., C. Blum (2005), "Ant colony optimization theory: A survey", Theoretical Computer Science 344 (2005) 243 – 278

Dorigo M., M. Birattari and T. Stutzle (2006), "Ant colony optimization," in IEEE Computational Intelligence Magazine, vol. 1, no. 4, pp. 28-39.

Duan, J., He, Z., & Yen, G. G. (2021). Robust multiobjective optimization for vehicle routing problem with time windows. IEEE Transactions on Cybernetics, doi:10.1109/TCYB.2021.3049635

Ehmke, J.F. and Campbell, A.M. (2014), “Customer acceptance mechanisms for home deliveries in metropolitan areas”, European Journal of Operational Research, Vol. 233 No. 1, pp. 193-207.

Ehmke, J.F. and Mattfeld, D.C. (2012), “Vehicle routing for attended home delivery in city logistics”, Procedia – Social and Behavioral Sciences, Vol. 39, pp. 622-632.

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

Fernie, J., Sparks, L. and Mckinnon, A.C. (2010), “Retail logistics in the UK: past, present and future”, International Journal of Retail & Distribution Management, Vol. 38 Nos 11/12, pp. 894-914.

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

Halvorsen-Weare, Elin & Fagerholt, Kjetil. (2010). Routing and scheduling in a liquefied natural gas shipping problem with inventory and berth constraints. Annals of Operations Research - Annals OR. 203. 1-20. 10.1007/s10479-010-0794-y.

He Z., G. G. Yen, and Z. Yi, (2019). “Robust multiobjective optimization via evolutionary algorithms,” IEEE Trans. Evol. Comput., vol. 23, no. 2, pp. 316–330.

Hertz, Alain, Eric Taillard, and Dominique De Werra(1995), "A tutorial on tabu search." Proc. of Giornate di Lavoro AIRO. Vol. 95.

Hildebrandt F. D., B. Thomas, and M. W. Ulmer, (2021). “Where the Action is: Let’s make Reinforcement Learning for Stochastic Dynamic Vehicle Routing Problems work!” arXiv.

Holland J.H. (1992), Genetic algorithms, Sci. Am. 267 (1) 66–73.

Hsu, C.-I., Hung, S.-F. and Li, H.-C. (2007), “Vehicle routing problem with time-windows for perishable food delivery”, Journal of Food Engineering, Vol. 80 No. 2, pp. 465-475.

Hu C., J. Lu, X. Liu, G. Zhang (2018), Robust vehicle routing problem with hard time windows under demand and travel time uncertainty, Computers & Operations Research, Volume 94, Pages 139-153

Jacobs-Blecha, C., & Goetschalckx, M. (1992). “The Vehicle Routing Problem With Backhauls: Properties and Solution Algorithms.” Georgia Tech Research Corporation.

Jalili S, Nallaperuma S, Keedwell E, Dawn A, Oakes-Ash L (2021) , Application of metaheuristics for signal optimisation in transportation networks: A comprehensive survey, Swarm and Evolutionary Computation, Volume 63:100865

Koski, J. (1984), Multicriterion optimization in structural design. In New Directions in Optimum Structural Design, edited by E. Atrek, R.H. Gallagher, K.M. Ragsdell and O.C. Zienkiewicz, pp. 485–503, (John Wiley and Sons: New York)

Koski, J. and Silvennoinen, R. (1987), Norm methods and partial weighting in multicriterion optimization of structures. Int. J. Numer. Methods Eng., 24, 1101–1121

Küçükoğlu, İ., Dewil, R., & Cattrysse, D. (2019). Hybrid simulated annealing and tabu search method for the electric travelling salesman problem with time windows and mixed charging rates. Expert Systems with Applications, 134, 279-303. doi:10.1016/j.eswa.2019.05.037

Laarhoven van, P.J.M., Aarts, E.H.L. (1987). Simulated annealing. In: Simulated Annealing: Theory and Applications. Mathematics and Its Applications, vol 37. Springer, Dordrecht.

Laporte, G. & Gendreau, Michel & Potvin, Jean-Yves & Semet, Frédéric. (2000). Classical heuristics for the vehicle routing problem. International Transactions in Operational Research. 7. 285 - 300. 10.1111/j.1475-3995.2000.tb00200.x.

Laporte, G., Toth, P., & Vigo, D. (2013). Vehicle routing: historical perspective and recent contributions. EURO J Transp Logist, 2 , 1-4. doi: 10.1007/s13676-013-0020

Lenstra J. K. and A. H. Kan (1981), “Complexity of vehicle routing and scheduling problems,” Networks, vol. 11, no. 2, pp. 221–227.

Li J. Q., P. B. Mirchandani, and D. Borenstein, (2007). “The vehicle rescheduling problem: Model and algorithms,” Networks, vol. 50, no. 3, pp. 211–229.

Li J. Q., P. B. Mirchandani, and D. Borenstein, (2009). “Real-time vehicle rerouting problems with time windows,” European Journal of Operational Research, vol. 194, no. 3, pp. 711–727.

Li Jiang, Mohamed Dhiaf, Junfeng Dong, Changyong Liang & Shuping Zhao (2020). A traveling salesman problem with time windows for the last mile delivery in online shopping, International Journal of Production Research, 58:16, 5077-5088, DOI: 10.1080/00207543.2019.1656842

Liu Ran, Xiaolan Xie, Vincent Augusto, Carlos Rodriguez (2013), “Heuristic algorithms for a vehicle routing problem with simultaneous delivery and pickup and time windows in home health care”, European Journal of Operational Research, Volume 230, Issue 3, Pages 475-486.

Ma, Q., Ge, S., He, D., Thaker, D.D., & Drori, I. (2019). Combinatorial Optimization by Graph Pointer Networks and Hierarchical Reinforcement Learning. ArXiv, abs/1911.04936.

Marla L., V. Vaze and C. Barnhart, (2018). “Robust Optimization: Lessons Learned from Aircraft Routing”, Computers and Operations Research, Vol 98, October 2018, pp. 165-184.

Marler R. Timothy & Jasbir S. Arora (2005), Function-transformation methods for multi-objective optimization, Engineering Optimization, 37:6, 551-570

Marler, R., Arora , J. (2004), Survey of multi-objective optimization methods for engineering. Struct Multidisc Optim 26, 369–395

Nasri, M., Metrane, A., Hafidi, I & Jamali, A. (2020). A robust approach for solving a vehicle routing problem with time windows with uncertain service and travel times. International Journal of Industrial Engineering Computations , 11(1), 1-16.

Nasser A. El-Sherbeny, Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods, Journal of King Saud University - Science, Volume 22, Issue 3, 2010, Pages 123-131, ISSN 1018-3647, https://doi.org/10.1016/j.jksus.2010.03.002.

Ochelska-Mierzejewska, J. (2020). Tabu Search Algorithm for Vehicle Routing Problem with Time Windows. In: Poniszewska-Marańda, A., Kryvinska, N., Jarząbek, S., Madeyski, L. (eds) Data-Centric Business and Applications. Lecture Notes on Data Engineering and Communications Technologies, vol 40. Springer, Cham. https://doi.org/10.1007/978-3-030-34706-2_7

Ogawara, S., Chen, J.C. and Zhang, Q. (2003), "Internet grocery business in Japan: current business models and future trends", Industrial Management & Data Systems, Vol. 103 No. 9, pp. 727-735.

Pan, S., Giannikas, V., Han, Y., Grover-Silva, E. and Qiao, B. (2017), "Using customer-related data to enhance e-grocery home delivery", Industrial Management & Data Systems, Vol. 117 No. 9, pp. 1917-1933.

Rahman S. and M. Rochan (2016), "A fast farthest neighbor search algorithm for very high dimensional data," 2016 19th International Conference on Computer and Information Technology (ICCIT), pp. 351-356

Rao, S.S. and Freiheit (1991), T.I., A modified game theory approach to multiobjective optimization J. Mech. Des., 113, 286–291.

Razmjooy N, Ramezani M (2018) Solution of the Hamilton jacobi bellman uncertainties by the interval version of adomian decomposition method. Int Rob Auto J 4(2):113 18.

Razmjooy N, Ramezani M (2019a) Uncertain method for optimal control problems with uncertainties using Chebyshev inclusion functions. Asian J Control 21(2):824–831 19.

Razmjooy N, Ramezani M (2019b) Interval structure of Runge-Kutta methods for solving optimal control problems with uncertainties. Computational Methods for Differential Equations 7(2):235–251 20.

Razmjooy N, Ramezani M (2021) Interval linear quadratic regulator and its application for speed control of DC motor in the presence of uncertainties. ISA Trans

Rijkswaterstaat. (2019, Juni). Rijksoverheid documenten/rapporten. Retrieved from Rijksoverheid: https://www.rijksoverheid.nl/documenten/rapporten/2019/07/15/rapportage-rijkswegennet-t1-2019

Salvesbergh M.W., (1985). "Local search in routing problems with time windows", Annals of Operations Research, 4, pp. 285-305

Savelsbergh, M. W. P. 1985. "Local Search in Routing Problems with Time Windows." *Annals of Operations Research* 4 (1-4): 285–305.

Scholten, J. (2021). Dynamic rerouting for an optimal delivery strategy.

Simacan. (2021, November). E-grocery growth since Covid. Amersfoort.

Slotboom, Cynthia (2019). The impact of soft time windows with penalties on the objective values of large real-world Vehicle Routing Problems. Delft University of Technology

Solomon, M. M. (1987). Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. Operations Research, 35(2), 254–265

Stacy A. Voccia, Ann M. Campbell, Barrett W. Thomas, (2013). The probabilistic traveling salesman problem with time windows, EURO Journal on Transportation and Logistics, Volume 2, Issues 1–2, Pages 89-107, ISSN 2192-4376, https://doi.org/10.1007/s13676-013-0018-0.

Sultana N. N., V. Baniwal, A. Basumatary, P. Mittal, S. Ghosh, and H. Khadilkar, (2021). "Fast Approximate Solutions using Reinforcement Learning for Dynamic Capacitated Vehicle Routing with Time Windows," arXiv e-prints, p. arXiv:2102.12088.

Suman, B., Kumar, P. (2006), A survey of simulated annealing as a tool for single and multiobjective optimization. J Oper Res Soc 57, 1143–1160

Tan, F., Chai, Z. -., & Li, Y. -. (2021). Multi-objective evolutionary algorithm for vehicle routing problem with time window under uncertainty. Evolutionary Intelligence, doi:10.1007/s12065-021-00672-0

Toth, P., Vigo, D. (2002). “Vehicle Routing Problem, SIAM Publishing, monographs on discrete mathematics and applications”, Philadelphia.

Vos Daniël. (2016). Basic principles of the traveling salesman. Delft: University Institute of Applied Mathematics.

Wang Dujuan, Jiaqi Zhu, Xiaowen Wei, T.C.E. Cheng, Yunqiang Yin, Yanzhang Wang (2019), Integrated production and multiple trips vehicle routing with time windows and uncertain travel times, Computers & Operations Research, Volume 103, Pages 1-12.

Yang, R.J., Tseng, L. and Cheng, J. (1994), Feasibility study of crash optimization, in Advances in Design Automation, Proceedings of the 1994 ASME Design Technical Conference, 20th Design Automation Conference, Minneapolis, MN, 2, September, 549–556

Yanik S., B. Bozkaya, R. deKervenoael, (2014). “A new VRPPD model and a hybrid heuristic solution approach for e-tailing”, European Journal of Operational Research, Volume 236, Issue 3, pp. 879-890.

Zhang Yu, Zhenzhen Zhang, Andrew Lim, Melvyn Sim (2021) Robust Data-Driven Vehicle Routing with Time Windows. Operations Research 69(2):469-485.

Appendices

Appendix A – Hub specific robustness formulas, based on historical data

Table 8: Robustness function data for all hubs

HubNR	HubID	Formula
		0.4072e^-0.062x
		0.2735e^-0.063x
		0.4485e^-0.078x
		0.2609e^-0.064x
		0.1299e^-0.084x
		0.1292e^-0.066x
		0.1342e^-0.074x
		0.3161e^-0.062x
		0.2086e^-0.061x
		0.3436e^-0.054x
		0.3161e^-0.062x
		0.4439e^-0.057x
		0.2821e^-0.068x
		0.2530e^-0.087x
		0.2264e^-0.084x
		0.1241e^-0.076x
		0.0598e^-0.072x
		0.3638e^-0.074x
		0.2046e^-0.077x
		0.3279e^-0.069x
		0.3161e^-0.062x
		0.3941e^-0.067x
		0.1553e^-0.070x
		0.2793e^-0.064x
		0.1398e^-0.067x
		0.4130e^-0.071x
		0.3796e^-0.064x
		0.2051e^-0.074x
		0.1109e^-0.078x
		0.2908e^-0.070x
		0.3148e^-0.076x
		0.4601e^-0.070x
		0.2883e^-0.072x
		0.1838e^-0.069x
		0.3860e^-0.037x
		0.1082e^-0.076x
		0.3161e^-0.062x
		0.3161e^-0.062x
		0.1434e^-0.073x
		0.2353e^-0.066x
		0.1982e^-0.062x
		0.3161e^-0.062x
		0.1603e^-0.050x

Appendix B – Distribution of all stops in relation to their planned minutes before the end of the corresponding time window

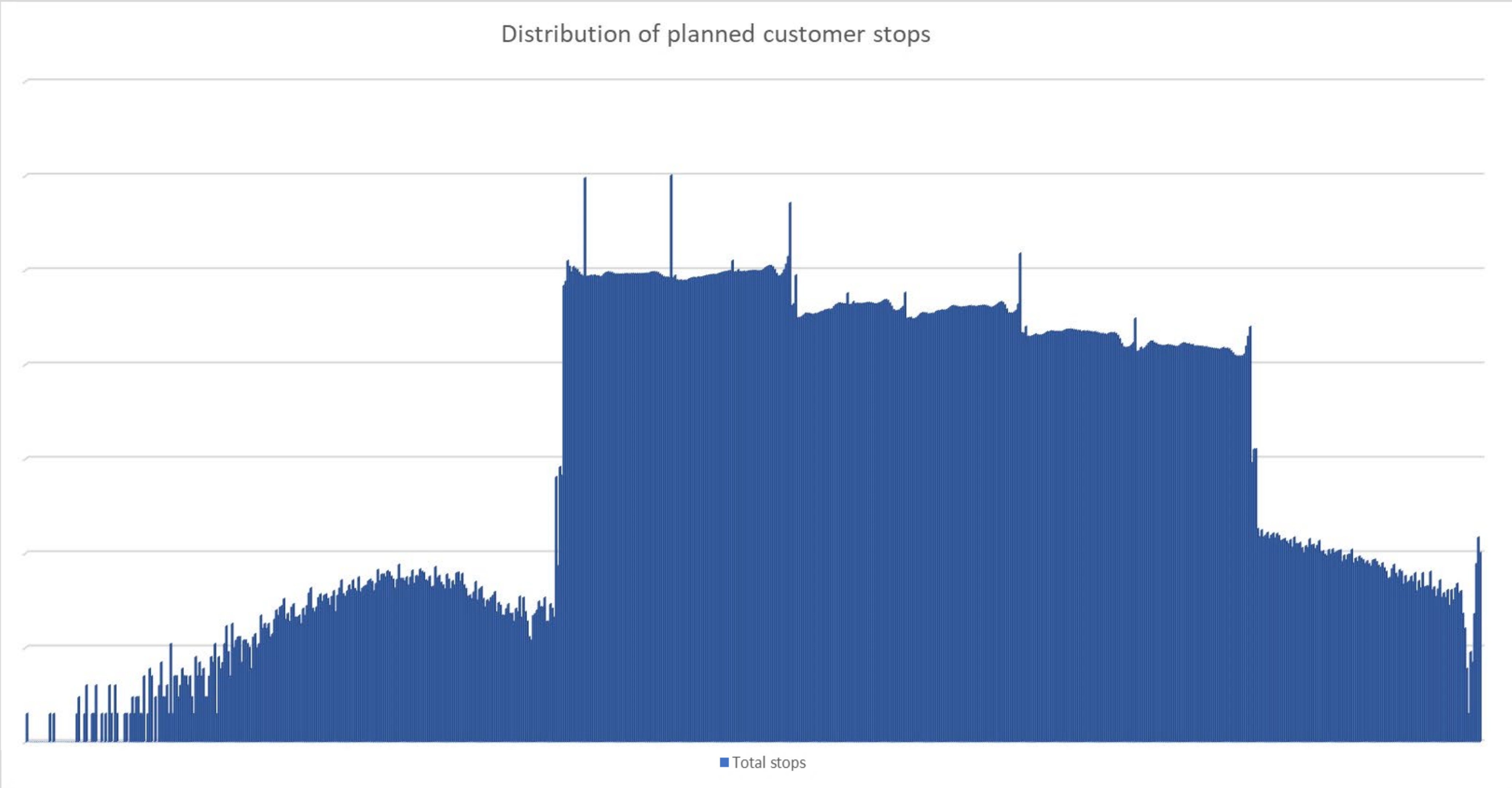


Figure 37: Distribution of number of customer stops in relation to the planned minutes before the end of the time window

Appendix C – Results before and after performing the rescheduling algorithm on two existing Trips

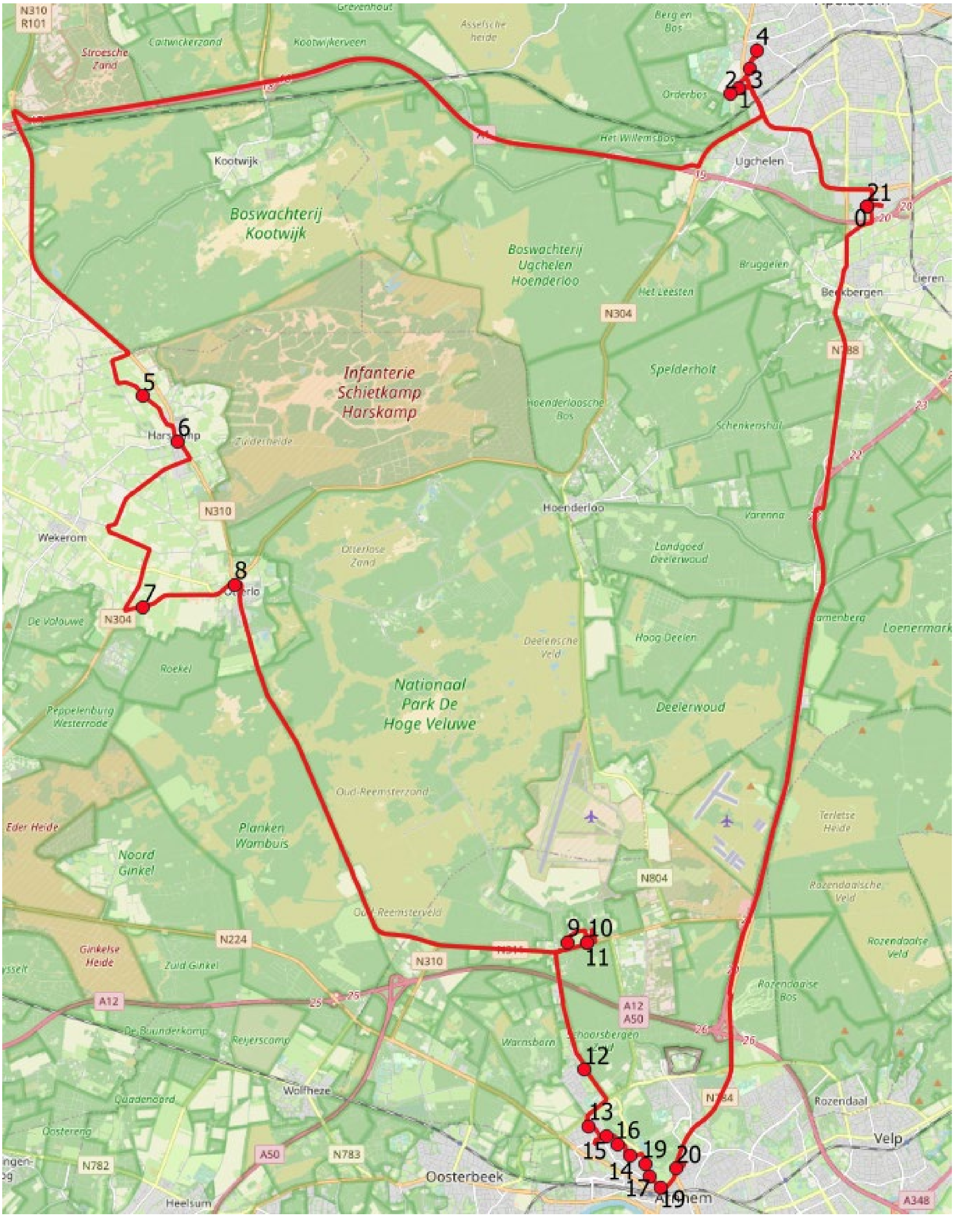


Figure 38: Old route before optimisation of Trip 2

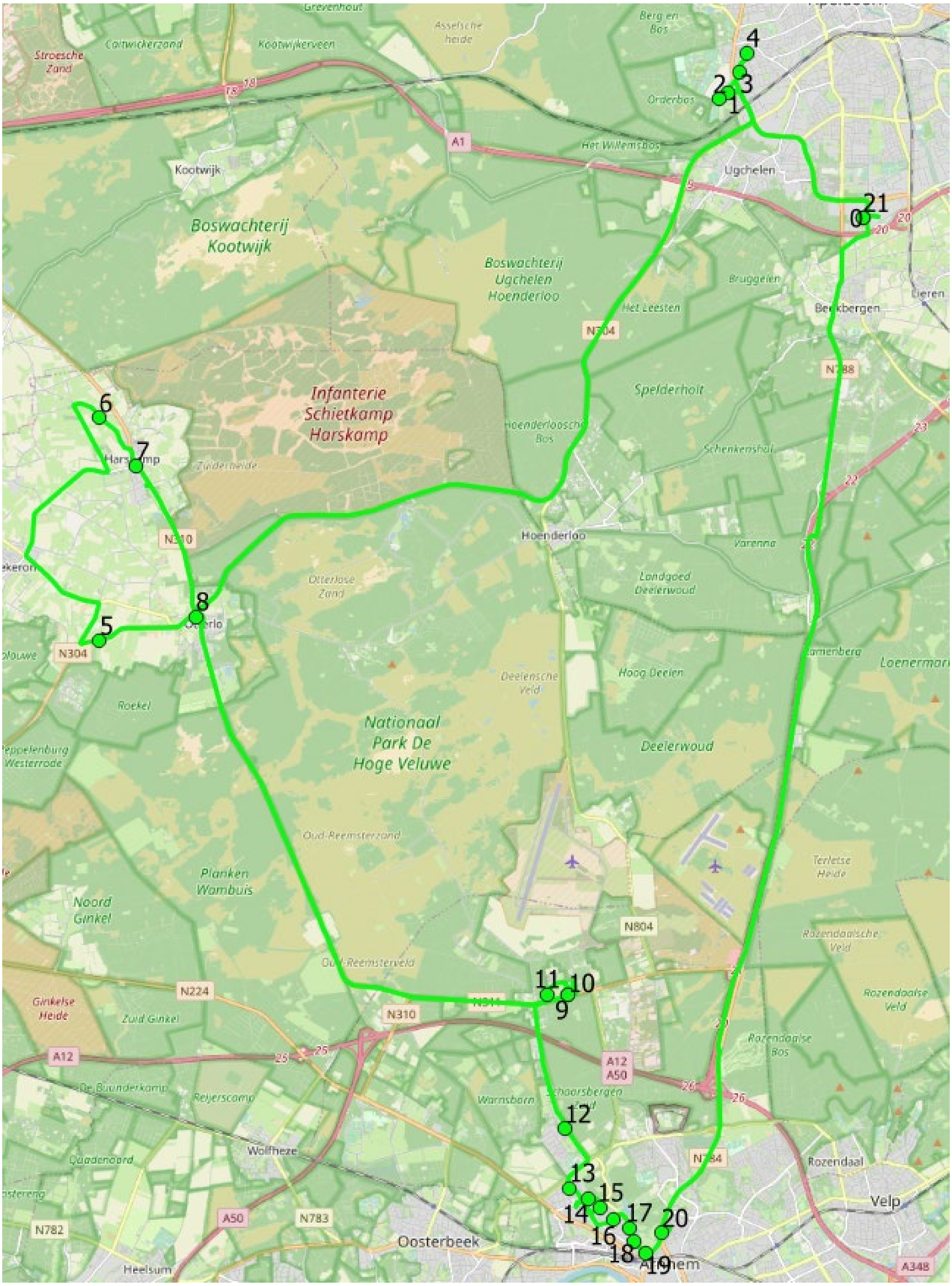


Figure 39: New route after optimisation of Trip 2

Appendix D – Example of the difference on control tower plan between old and new plan

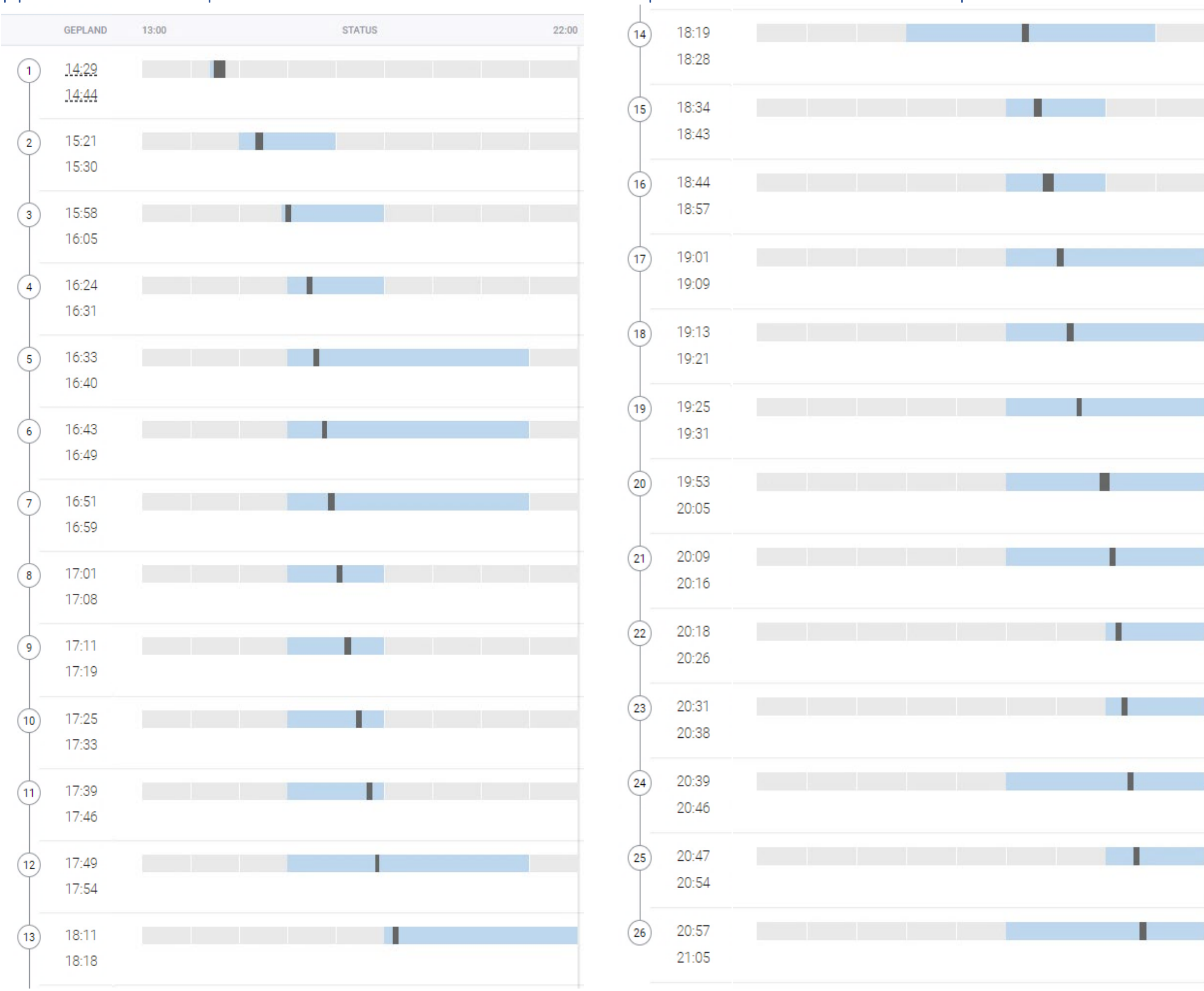


Figure 40: Example of an original schedule before optimising

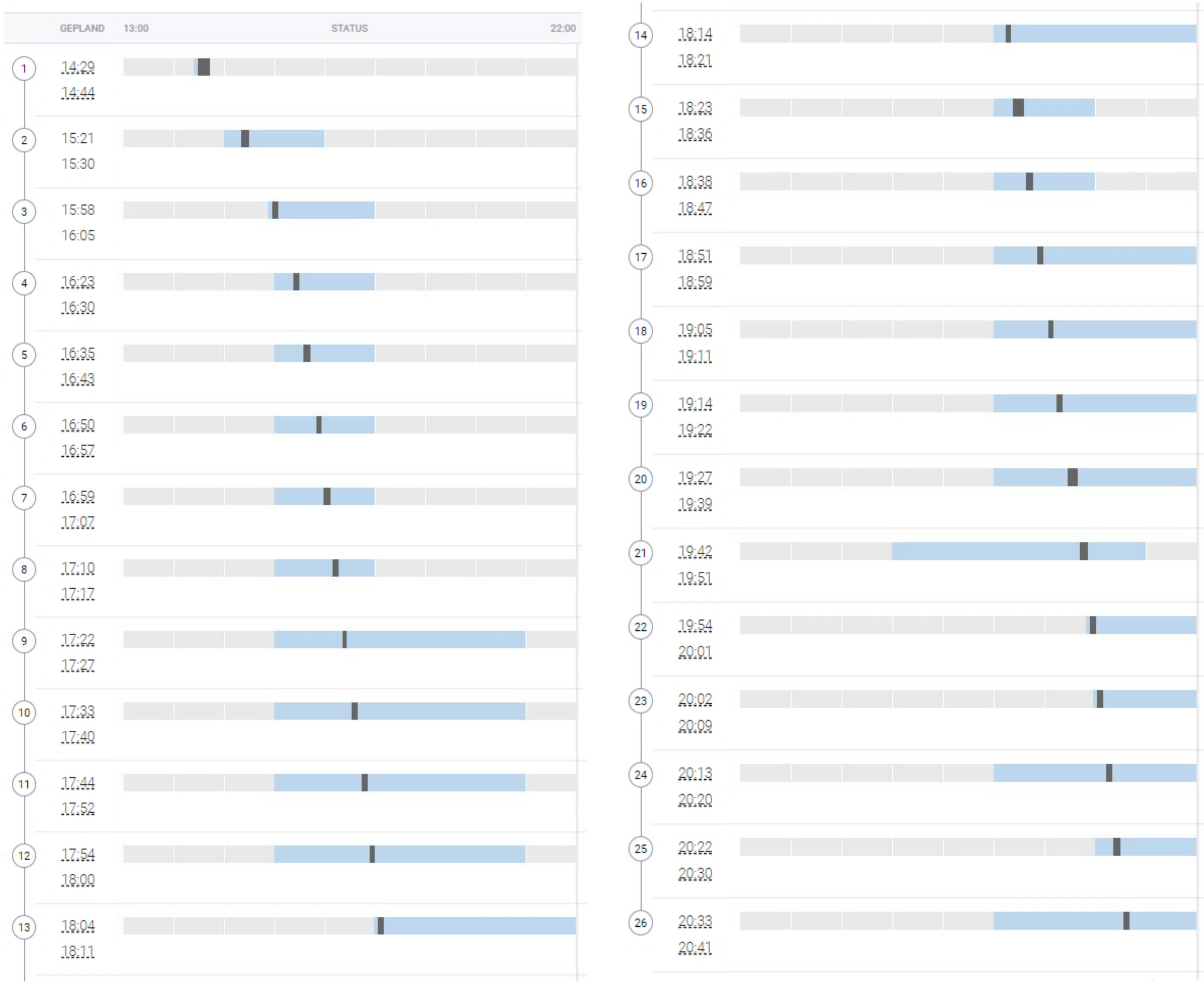


Figure 41: Example of an optimised schedule after performing robustness improvement (based on Figure 40)