MASTER

Increasing delivery efficiency by Cargo Hitching

a case study

Bakker, J.

Award date:
2015

Link to publication
Eindhoven, May 2015

Increasing delivery efficiency by Cargo Hitching: A case study

by
Jaap Bakker

BSc. Industrial Engineering and Innovation Sciences – TU/e 2012
Student ID: 0637097

In partial fulfillment of the requirements for the degree of

Master of Science
In Operations Management and Logistics

Supervisors:
Dr. Luuk Veelenturf Eindhoven University of Technology
Prof. dr. Tom van Woensel Eindhoven University of Technology
TUE. School of Industrial Engineering.
Series Master Theses Operations Management and Logistics

Subject headings: Cargo Hitching, Pickup & Delivery Problem, Case Study, Multi Request Type, Multi Depot, Multi Vehicle, Transshipments, Scheduled Lines, Local Search, Hill Climbing, Tabu Search, Large Neighborhood Search, Mixed Integer Linear Program.
# Contents

**CHAPTER I MANAGEMENT SUMMARY** ........................................................................................................... 5

**CHAPTER II ABSTRACT** ................................................................................................................................. 8

**CHAPTER III PREFACE** ................................................................................................................................. 9

**CHAPTER 1 INTRODUCTION** ......................................................................................................................... 10

1.1 OVERVIEW ................................................................................................................................................... 10

1.2 DESCRIPTION OF CARGO HITCHING ................................................................................................. 11

1.3 THESIS OUTLINE ..................................................................................................................................... 11

**CHAPTER 2 LITERATURE REVIEW** ............................................................................................................. 13

2.1 APPROACHES FOR SOLVING THE STANDARD PDP ................................................................................ 13

2.2 EXTENSIONS OF THE PDP ...................................................................................................................... 16

2.3 CONCLUSIONS ........................................................................................................................................ 19

2.4 CONTRIBUTION ....................................................................................................................................... 19

**CHAPTER 3 LEKKERLAND CASE STUDY** .................................................................................................... 20

3.1 A GENERAL DESCRIPTION ....................................................................................................................... 20

3.2 SCOPE OF CASE STUDY ........................................................................................................................ 20

3.3 RESEARCH QUESTIONS ........................................................................................................................... 22

3.4 PROBLEM DESCRIPTION AND ASSUMPTIONS ..................................................................................... 23

**CHAPTER 4 EXACT FORMULATIONS** .......................................................................................................... 25

4.1 NOTATION .................................................................................................................................................. 25

4.2 MATHEMATICAL FORMULATION .......................................................................................................... 28

4.3 TIGHTENING OF DECISION VARIABLES ............................................................................................. 32

4.4 COMPARISON TO GHILAS ET AL. ........................................................................................................... 34

**CHAPTER 5 PDP-SL METAHEURISTIC** ........................................................................................................... 35

5.1 ALGORITHM PHASES ............................................................................................................................... 36

5.2 FIRST PHASE: INITIALIZATION TRUCK ROUTES .................................................................................. 37

5.2 SECOND PHASE: INITIALIZATION TAXI ROUTES .................................................................................. 37
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3</td>
<td>THIRD PHASE: INSERT SCHEDULED LINES</td>
<td>38</td>
</tr>
<tr>
<td>5.4</td>
<td>FOURTH PHASE: IMPROVE FOUND SOLUTION</td>
<td>39</td>
</tr>
<tr>
<td>5.5</td>
<td>COMPARISON TO GHILAS ET AL.</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td>CHAPTER 6 COMPUTATIONAL RESULTS</td>
<td>43</td>
</tr>
<tr>
<td>6.1</td>
<td>EXPERIMENTAL DESIGN</td>
<td>43</td>
</tr>
<tr>
<td>6.2</td>
<td>EXPERIMENTAL RESULTS</td>
<td>44</td>
</tr>
<tr>
<td>6.3</td>
<td>EVALUATION OF HEURISTIC</td>
<td>53</td>
</tr>
<tr>
<td></td>
<td>CHAPTER 7 CONCLUSIONS AND FUTURE WORK</td>
<td>55</td>
</tr>
<tr>
<td>7.1</td>
<td>SUB RESEARCH QUESTIONS</td>
<td>55</td>
</tr>
<tr>
<td>7.2</td>
<td>GENERAL CONCLUSION</td>
<td>56</td>
</tr>
<tr>
<td>7.3</td>
<td>LIMITATIONS</td>
<td>58</td>
</tr>
<tr>
<td>7.4</td>
<td>FUTURE WORK</td>
<td>58</td>
</tr>
<tr>
<td></td>
<td>CHAPTER 8 BIBLIOGRAPHY</td>
<td>60</td>
</tr>
<tr>
<td></td>
<td>CHAPTER 9 APPENDIX</td>
<td>63</td>
</tr>
</tbody>
</table>
Chapter i
Management Summary

This report is the results of a master thesis project, executed partially at Lekkerland, Son. Lekkerland is a multinational wholesaler and partner for all retail formats that give passengers convenient “on-the-go” consumption. The location Son has the biggest distribution center (DC) of Lekkerland in the Netherlands and contains a large fleet of heterogeneous vehicles. From this distribution center the Lekkerland fleet transports cargo to Noord-Brabant, Limburg, Zuid-Holland, Utrecht and Gelderland as well as cross-docking to distribution centers in Waddinxveen and Meppel.

Problem Statement

The current transportation process performed by the Logistic Service Provider (LSP) Lekkerland is done in a traditional manner. They have a certain fleet of trucks to their disposal and they use this fleet to deliver cargo towards their clients. The Cargo Hitching forum (Dinalog, 2015) has proposed new ways to outsource part of the delivery process. This new way of outsourcing uses public transportation modes which have a surplus in capacity. This overcapacity can be used to transport cargo. Cargo Hitching long-haul implementations can already be found in the real world, for example cargo transportation by passenger airplanes which have additional capacity. In short-haul however this is less common. The problem which this thesis has investigated is the integration of this Cargo Hitching concept into the current delivery system of Lekkerland. A case study is created in order to investigate if this integration can be successful and if it can induce benefits for multiple stakeholders. A graphical illustration of this concept can be seen in Figure 1.

Case study

To assess the potential of the Cargo Hitching concept for the LSP, I have chosen to look at the historical data of the LSP its clients in Eindhoven and Helmond. To be more precise, a dataset has been retrieved of week 28 in 2014 which contains data about the LSP clients and their orders. The scope of this study involves bus lines and taxis for the integration of Cargo Hitching into the current LSP delivery system. This choice has been made based on available passenger transport modes in Eindhoven and Helmond. The bus lines are used to transport the cargo orders from the depot of the LSP to the central station of Eindhoven and Helmond. From these location the last mile will be performed by the taxis. This new system holds that the LSP has the option to let their cargo order travel from their depot to a transshipment hub with a LSP truck, from there it will travel to a central station with a bus and from there to its final destination the order will travel with a taxi. Taxis still serve their own passengers but if they have time and capacity left then they can additionally carry cargo.
The case study involves multiple stakeholders. The first stakeholder is the LSP which performs the current service of delivering cargo to its clients. These delivery orders will be called cargo requests from now on. The LSP wants to deliver these cargo requests at minimal cost, but for example has to follow certain time windows which are imposed by the clients themselves. For the LSP the goal of the Cargo Hitching integration is therefore to minimize their delivery cost.

The second stakeholder is the bus line company. They have a surplus in capacity and can therefore transport cargo with minimal additional cost. The bus line company wants to maximize the use of its available capacity and in return receive a financial compensation. The third stakeholder is the taxi company. They also have a surplus in capacity at times and with this capacity they can perform the last mile of the cargo request. The taxi company its goal for the Cargo Hitching is to receive financial compensation for their additional kilometers travel. The fourth stakeholder is the society. Trucks impose quite some disturbances on the road. They cause for example air pollution, sound pollution and road congestion. The goal of the Cargo Hitching concept for the society is to reduce the number of trucks on the road. Figure 2 first shows the current simplified delivery by truck and Figure 3 shows the proposed delivery with the integration of Cargo Hitching.

**Methodology**

The overall goal of the case study was to perform experiments with multiple datasets and create routing solutions for both the current LSP delivery system and for the proposed Cargo Hitching delivery system. These routing solutions have been compared and based on this comparison I was able to assess the potential of the system for all four of the stakeholders.

The routing solution must be possible in the real world. A mathematical formulation for both the current LSP delivery system and the Cargo Hitching delivery system is created in order to control if they are applicable to the real world. These mathematical formulations are a generalization of the real world and its limitations. The created mathematical formulations are mixed integer linear programming (MILP) formulations because their decision variables can contain as well integer as continuous values and all of the constraints and functions must be linear. A routing solution is feasible if all the constraints in the MILP formulation are met.

The next challenge was the creation of these solutions. A routing solution has been created for the current LSP delivery system with an exact method. An exact method creates the optimal solution, if possible,
but is a relatively slow method. The proposed Cargo Hitching delivery system is more complex than the current delivery system and I was unable to create an optimal solution for a large enough dataset due to computer memory problems. In order to create a good solution for the Cargo Hitching delivery system I have created a heuristic method. This heuristic method does not always provide the optimal solution but provides a good solution within a faster time. The heuristic method uses greedy elements for the initialization phase. I uses, among others, large neighborhood and tabu search elements for the improvement phase. The resulting routing solution of this heuristic method for the Cargo Hitching system can be compared to the exact routing solution of the current delivery system in order to assess the potential of the Cargo Hitching concept in this case study.

**Results**

The numerical results show that there is a potential benefit for all of the stakeholders involved. In the results the benefit for all the four stakeholders is evaluated under different parameter settings. Parameters settings are the different values for, for example, the cost of a truck per kilometer, the cost of a taxi per kilometer and the cost for a trolley to travel with a bus line. The value of the parameters has an influence of the usage of Cargo Hitching by the LSP and above €1 - €2 for the usage of the service of a bus line the results tend to show that it is not beneficial for the LSP to use the service of a bus line.

![Figure 4 Solution to the current LSP delivery system](image1)

![Figure 5 Solution to the new Cargo Hitching delivery system](image2)

**Conclusion and recommendations**

The case study shows that it is possible to create routing solutions which benefit the stakeholders but that there are also multiple assumptions which have to be considered first. Some of the assumptions made in this case study can results in additional cost. Also investment have to be made before this system can be implemented. Investment include changes to the infrastructure, vehicles and information systems. The best way to investigate these changes and assumptions is to launch pilots to test the Cargo Hitching in the real world. Another recommendation is to look at areas which do not have a high density of delivery locations. The higher the density of deliveries in areas, the more scheduled lines and taxis have to be used in order to create benefit for the system.
Chapter ii

Abstract

In this thesis I will test a relatively new concept called Cargo Hitching which is a specialization of the pickup and delivery problem. This concept allows cargo to switch between vehicles and scheduled services like a bus line. The experimenting is based on a case study which uses real world data of a logistic service provider (LSP) company. I attempted to recreate the real world as close as possible and chose to extract the travel durations from Google Maps and for example that vehicles can be too big for certain streets. The potential of the system has been tested by a heuristic algorithm for the Cargo Hitching system. This algorithm is able to improve optimal routing solutions without transshipments and scheduled lines by multiple heuristics which use element of the hill climbing method, tabu search and large neighborhood search. The algorithm is also able to create a solution from scratch. The objective cost function is the total cost for the routing of trucks, taxis and scheduled lines in order to service the cargo and passengers. The goal of the exact and heuristic methods is to minimize this objective cost function. The performance of the heuristic algorithm is tested by an optimal solution to a mixed integer linear problem formulation of the current LSP system as lower bound and a mixed integer linear problem formulation for the desired Cargo Hitching concept as upper bound.

Keywords: Cargo Hitching, Pickup & Delivery, Case Study, Multi Request Type, Multi Depot, Multi Vehicle, Transshipments, Scheduled Lines, Local Search, Hill Climbing, Tabu Search, Large Neighborhood Search, Mixed Integer Linear Program.
Chapter iii
Preface

This thesis is written for the completion of the master program Operations Management and Logistics at the Eindhoven University of Technology. In the past half year I have researched the pickup and delivery problem and a more complex variation which closely resembles the real world case of a logistic service provider (LSP). The LSP headquarters was located in Son, Eindhoven. I would especially like to thank my first supervisor from the Eindhoven University of Technology, Luuk Veelenturf, and also my second supervisor from the Eindhoven University of Technology, Tom van Woensel. Luuk Veelenturf has given me a lot of valuable feedback, advice and support during the project. Tom van Woensel introduced me to the concept of Cargo Hitching and set up the communication with the LSP. I would also like to thank the LSP to give me a place where I could gain a lot of valuable data and information about the delivery process. Last but not least I would like to thank my friends, family and partner for the support during the project.
Chapter 1
Introduction

The field of route optimization focuses on finding the best route between certain points. In this field there are a lot of possibilities for improvements and extensions. A large research subject within this field is the concept of City Logistics. A certain set of city logistic solutions aims to reduce the nuisances associated to transportation in urban areas while still supporting the economic and social development of the cities (Crainic, Ricciardi & Storchi, 2009). In city logistics one of the opportunities is that the capacity utilization of for example busses, trains and subways is not used to its full extend. This is especially true when looking at time windows outside the peak hours. A potential solution to this opportunity is for example lowering the prices for traveling with trains outside the peak hours or increase the prices of parking your car in the city center. Both of these solutions offer a financial incentive for passengers to make use of the public transportation and also an incentive for traveling outside of the peak hours. Another option is to create incentive for cargo to travel with for example busses when the capacity utilization is not yet optimal. In order to investigate these possibilities we need an extension of the standard route optimizations.

This thesis focuses on the testing of this possible extension to current route optimizations by allowing route solutions where transfers between vehicles are possible and where parts of the cargo route can involve scheduled services like busses, trains or more. I will explore what the impact will be on the cost for the logistic service provider (LSP), public transporters and possible benefit to the society. The proposed system will be modeled as a Pickup and Delivery Problem with Scheduled lines (PDP-SL), what this exactly means will be showed in chapter 2.

1.1 Overview

In the literature there has been recent research on the possibilities of letting cargo travel with public transportation companies, see Ghilas et al. (2015) and Li et al. (2014). They have shown that a synergy may exist when logistic service providers are cooperating with scheduled service providers.

The scheduled services currently have a challenge in the Netherlands to increase the utilization of public transport because many citizens perceive benefit from traveling with their own transportation modes instead of scheduled services. A different approach is one which does not mainly focus on increasing the incentive for citizens to use scheduled services but instead tries to increase the incentive for cargo transportation companies to make use the scheduled services when there is unused capacity in the scheduled services.

The general idea is that we can combine for example the passengers and cargo in order to achieve an increase of the used capacity in public transportation. The main question is:
Can cargo transportation be integrated into passenger transportation and achieve a reduction in total cost and impact on the environment, and how can this be achieved?

To achieve an efficient working system which can be implemented in the real world we first need a solid framework which can be analyzed and tested with data out of the real world. Researchers already proposed frameworks and models for parts of this Cargo Hitching problem. Other research on this topic is for example performed by Ghilas et al. (2014) and Li et al. (2014), but has not yet been tested with real life data and constraints.

1.2 Description of Cargo Hitching

The possible passenger transportation modes for this concept include trains, buses, trams, subways, airplanes, ferries, taxis and even more. Cargo Hitching is already being deployed at some places in the world. Most of these integrations have the purpose for long-haul transportation, a nice example is the air transportation. The excess capacity, which is not used by passenger, can be sold for transporting cargo at a low price. Another example is the Norwegian Hurtigruten which can carry cargo and people at the same time (Hurtigruten, 2014). The Hurtigruten is a large ship in Norway which travels from shore to shore and acts as a cruise ship for its passengers. At the same time it serves as a transportation ship for cargo.

Short haul, for example intra- and intercity passenger transportation, however is a less used category for Cargo Hitching. The Cargo Hitching model that I will investigate for this case study is one of these short haul ideas. Modes of transport like boats, trains or buses are an example of a scheduled service and follows a predetermined route and departure time. If they carry passengers then they will not deviate from their original scheduled route to transport a passenger to its final destination. The passenger still has to be picked up somewhere in the route and delivered to its destination by another mode of transport. A train for example will not deviate from its original routing plan or departure times in order to bring a passenger home. These scheduled lines are an interesting research topic in the route optimization planning domain.

A challenge in using scheduled services as possible transportation modes for short-haul cargo delivery is to get the cargo to its final destination. An additional transportation mode which can be used in models with scheduled services can be passenger taxis. These taxis can carry out the last mile of a request. More information on this idea of using scheduled lines as well as taxis for the delivery of cargo can be found in Ghilas et al. (2013).

1.3 Thesis outline

In this section I will provide an outline of this thesis. Chapter 2 is devoted for the discussion on relevant literature to the problems discussed in this thesis. In this literature study I will also present the basic concepts of the Pickup and Delivery problem (PDP) and expand this to the Pickup and Delivery problem with Transshipments and Scheduled Lines. Section 2.1 show the popular and most promising solution
methods for the PDP, this includes exact and heuristic methods. Section 2.2 shows the most relevant extensions or variants of the original PDP. Section 2.3 gives a conclusion about the performed literature study with current trends and history. In section 2.4 I will elaborate on what my contribution will be to the literature.

Chapter 3 presents the case study which will be the main focus of my thesis. In section 3.1 I give a description of the LSP which I have chosen for the data and more information about their location. The scope of the study will be discussed in section 3.2 and also why I chose this scope. Section 3.3 will follow up on this scope and presents the research questions which I will answer later on in the thesis. In section 3.4 I discuss the assumptions and special constrains which I have made in order to being able to model the case study with a mathematical formulation.

Chapter 4 is a presentation of the mathematical formulations created of the two systems, one without scheduled lines and one with scheduled lines. Section 4.1 starts with the general notation which will be used for the formulation. Section 4.2 shows the two different mathematical models and gives additional explanation about the constraints and objective function. The decision variables in the model will be tightened to reduce the complexity and this will be shown in section 4.3. In Section 4.4 I have compared my formulations with the formulations of Ghilas et al. 2015.

Chapter 5 presents a heuristic which first creates an initial solution for the PDP-SL and second improves this method with multiple improvement heuristics. Section 5.1 gives an outline of the phases of the heuristic which I have used to test the case study and sections 5.2 to 5.4 zoom in on the phases described in section 5.1. In section 5.5 I have compared my heuristic with the heuristic created by Ghilas et al. 2015.

Chapter 6 presents the computational results which were generated by the methods explained in chapter 4 and 5. Section 6.1 gives an outline of the experiments which were held in order to retrieve the results. More information is given here about the size of the data, number of test batches, which parameters were used and what software/hardware was used. Section 6.2 provides the results from the experiments from section 6.1 and shows my interpretation on these results. Section 6.3 is used to determine the quality and performance of the used heuristic in contrast with the exact methods form section 4.2.

Chapter 7 provides the conclusion and limitations of my thesis and future research opportunities. Section 7.1 answers the sub research question from the research design. Section 7.2 provides general conclusions about the case study. Section 7.3 presents the limitations of my thesis research. Section 7.4 presents my thought on future research topics. Chapter 8 contains the bibliography. Chapter 9 contains the appendices.
Chapter 2
Literature review

The models used for the case study are based on the Pickup and Delivery Problem (PDP). In the PDP a set of requests needs to be moved from their pickup location to their delivery location. These requests can be moved by a set of vehicles. The goal is to create an optimal routing solution in which all of the requests have been moved correctly at the lowest cost. The case study involves pickups and deliveries as service. Cargo and passengers first have to be picked up by a vehicle and later will have to be dropped off at their destination location. If the case study only involved the cargo delivery by the logistic service provider (LSP) because the all of the pickups are done in their depot then I could have also chosen to use a Vehicle Routing Problem (VRP). The VRP needs to deliver a set of requests to their delivery location with a set of vehicles which both start at the depot. The proposed system includes vehicles which have to pick up for example the passengers before they can drop them off at their destination location, therefore I chose to use the PDP as foundation for my own model. For this reason I chose to mainly focus on the PDP in the literature review and less on the Vehicle Routing Problem.

Many variations of the PDP have been studied over the last decades. The generalized problem is a problem with characteristics that return in all of these different variations of the PDP. Savelsbergh and Sol (1995) describe this generalized version of the pickup and delivery problem. They define the generalized PDP ‘to consist of a set of transportation requests which are serviced in a number of routes. A route starts at a given node and ends at another given node, which may be the same, and is driven by one vehicle $k$. Between the start and end node a number of pickup and a number of delivery nodes are included in the route. For every delivery node there is a corresponding pickup node. All of the nodes are served only once by a vehicle $k$. During a route the vehicle load never exceeds its capacity. The whole solution is a set of routes, for which it holds that every pickup and delivery node is visited exactly once. They also consider a number of more specific problems, including the pickup and delivery problem with time windows.’

2.1 Approaches for solving the standard PDP

Multiple type of methods exist in literature for creating a solution to the PDP. In the next two sections I will give a short overview of the existing literature on this subject. The first section reviews the current literature on exact methods. Exact methods are tools to find the optimal solution to a certain problem. The second section reviews the current literature on heuristic methods. Heuristic methods are tools which follow a certain guideline in order to find a solution. They cannot guarantee that they provide the optimal solution but in return are generally faster.

2.1.1 Exact methods

Exact methods have been a popular research domain for already a few decades. In 1981 one of the first successes were found with an exact algorithm by Christofides et al. (1981). They created the Dynamic
Programming with State Space Relaxation method and the Branch-and-Bound method which could solve up to 25 requests. Laporte et al. (1985) created the Branch-and-Cut method which could solve up to 60 requests. Since then new variants on this Branch-and-Cut methods were created, for example Baldacci et al. (2008) created a variation called the Branch-and-Cut-and-Price method which was used to create an optimal solution for up to 121 requests.

The standard PDP has been proved to be NP-hard by Lenstra and Rinnooy Kan (1981). NP-hard problems are very hard to solve with brute force exact methods. CPU time and required RAM increase exponentially when the size of the problem grows. Real life problem take a very large amount of CPU time when trying to use an exact method for an optimal solution to the PDP-SL. Relatively small sets can still take up to days of running. Many researches also require multiple different scenarios to be solved and analyzed in relative short amount of time.

For both real world companies and researchers a brute force exact method like Branch-and-Cut are not a good option. Some researchers try to decrease the CPU time with techniques as the Column Generation (Desaulniers, Desrosiers, & Solomon, 2005).

The column generations splits the original mixed integer linear programming formulation in two parts. A mixed integer linear programming is a mathematical formulation which models a certain scenario or case study. This model has an optimization function which assesses the performance based on integer and continues decision variables. The goal of the model can be to maximize or minimize this value. Unfortunately the column generation only works under certain assumptions which are not applicable to all models, including the PDP-SL. For example the column generation assumes that a cargo request will be served by only one vehicle which is not the case in the PDP-SL.

2.1.2 Metaheuristics

As mentioned before, when a research requires an exhaustive search then it will probably not be possible to look at large datasets. For these problems there is an alternative type of method called heuristic or metaheuristic methods. (Meta)heuristics can be described as a set of rules which are used to search for a feasible solution. Following these set of rules to create a solution will most of the times be significantly faster than exact methods, the downside is that it cannot guarantee to find the optimal solution because it does not search the entire solution space. Most of these (meta)heuristics require an initial solution. These initial solutions can quickly be generated by for example a construction heuristic which uses elements from the greedy method. This initial solution is created by starting with an empty solution and then searching in the neighborhood space. The neighborhood space is the set of solutions which can be reached by a single modification to the current. This algorithm will repeat the process of looking in neighborhoods until a feasible routing solution is created.
These modifications can be of different types throughout the search. The modification is determined by an operator which is a predefined method for making changes to the current solution. With the help of these modifications the search space can be explored.

In the last 20 years the metaheuristic have become more popular. A more general description is that a metaheuristic is a higher-level procedure or heuristic designed to find, generate, or select a lower-level procedure or heuristic. One of the first metaheuristics is the tabu method. The first tabu implementation was the tabu search (Willard, 1989) and after that implementation many new metaheuristics were developed. Other successful metaheuristics implementations worth mentioning are the Simulated Annealing searches (Osman, 1993), Adaptive Memory (Rochat & Taillard, 1995), Ant System optimization (Reimann et al., 2004), Adaptive Very Large Neighborhood Search (Pisinger & Ropke, 2007) and of the most recent GRASP and Evolutionary search (Prins, 2009). Below I will present a few of the most popular metaheuristic concepts.

**Hill climbing.** This is one of the most basic methods used for improving an initial solution (Tovey, 1985). The rules of this method tell the system to look for new solutions from the neighborhood solution space and accept one of these when there is an improvement of the current solution. The most obvious downside of this method is getting stuck in a local optimum. Therefore the end solution is highly dependent on the initial solution. An easy possible remedy for the problem of initial solution is to restart the metaheuristic with a different initial solution and to see if this brings an improvement.

**Tabu search.** Tabu search is a method which keeps track of previously visited solution. Just as the Hill Climbing method it searches the neighborhood and uses the best solution in that neighborhood if it is not on the tabu list (Glover, 1989). This process is done until no more improvements can be found and then the process repeats but with the tabu list created in previous rounds. One great example is a method based on the tabu search created by Cordeau and Laporte (2003). They created a method for the Dial-A-Ride-Problem (DARP) which is a variant of the original PDP. Additionally they added a flexibility which allowed for infeasible temporary solutions which inquire a penalty. Further on in the method these penalties will increase and become too costly.

**Simulated Annealing.** Simulated Annealing is a method which tries to prevent the early narrowing of the search space (Kirkpatrick et al, 1983) (Zidi et al., 2012). The method keeps track of a certain temperature which start of high and gets lower as the algorithm runs. The higher the temperature the easier a solution is accepted. This way it prevents the early narrowing of the search space. The temperature lowers according to a certain equation and therefore the required fitness level rises and worse solutions do not get accepted any more. An example of implementation is done by Sahin et al. (2012) for the Pickup and Delivery with Time Windows and Split Loads. They create a method which is a mix between simulated annealing and tabu search. They do not choose the best solution in the neighborhood but they select the solution in the neighborhood with its own best neighborhood. The tabu
part of this method makes sure that this best solution is not already on the tabu list. If the solution with the best neighborhood is worse than the current solution then it depends on the current temperature if it is selected.

**Large Neighborhood search.** Large neighborhood search (LNS) is a relatively new but frequently mentioned method in literature. Instead of having neighborhoods which are relatively small because they are based on one change according to an operator the LNS has larger neighborhoods which can be reached by changing a larger part of the solution. This reduces the chances of getting caught in a local optimum. Ghilas et al. (2014) adapted this LNS and created an Adaptive LNS for the PDP-SL. This adaptive approach means that the improvement history of all the operators defines the chance of using them again. Another implementation example is done by Ropke and Pisinger (2006) which also used an adaptive approach in combination with simulated annealing. They use a temperature as in simulated annealing to determine the chance that a solution made by an operator is accepted.

### 2.2 Extensions of the PDP

#### 2.2.1 Scheduled lines and Transfers

The integration of scheduled services is also an important variant of the PDP for the route optimization field. The term scheduled services refers to the fact that the transport mode cannot deviate from their original route and has predetermined departure times. Aldaihani and Dessouky (2003) developed a system that integrates scheduled lines with a general PDP creating what the authors call a hybrid routing problem. This problem incorporates the possibility of transfers between scheduled lines. It can best be compared with a Dial-a-Ride problem (DARP) but also incorporating transfers between two or more different modes of transport per customer. The goal of the integration is to reduce the vehicle miles of the on-demand pickup and delivery vehicles while not significantly reducing the customer service level. They propose a heuristic algorithm and test the performance using data from a transit agency. For the creation of an initial solution they use an insertion procedure with three phases. In the first phase, all the candidate routes/paths that meet a certain criterion for each request are identified. In the second phase, an initial solution is created by identifying a feasible path from the candidates’ list that has the shortest on-demand vehicle distance. In the third phase, the initial solution is fed into an improvement procedure. In this procedure, they try to identify an alternative path that reduces the total passenger trip time for requests that have multiple hybrid paths. This solution from the insertion/improvement procedure is the initial solution in the Tabu Search. The Tabu Search consists of two function Re-Sequencing and Re-Assigning. With this method they solve up to 155 requests per day.

The work of Quadrifoglio et al. (2006) involves the creation of a Mobility Allowance Shuttle Transit (MAST) service. This is a semi scheduled service as it uses timetabled travel line from one location to another but can deviate slightly from this path. The research involves looking for the best possible bounds for this new service. Hickman and Blume (2001) developed an approach for the creation of a planning for customers and pickup and delivery vehicles in the situation where the customers use pickup
and delivery vehicles to transfer to and from a scheduled line. They solve this problem using insertion heuristics. Horn (2002) proposes an insertion heuristic algorithm for a system which uses transfers between multiple journey legs, e.g. walking, bus, metro, taxi. Based on this algorithm they created a software programming for the scheduling of customers. The program is called LITRES-2 and exists of multiple modules. The heuristics uses the control modules which plan the journeys. The objective function minimizes the cost for the passengers.

Cortés and Jayakrishnan (2002) investigated the system of High Coverage Point to Point Transit (HCPPT). High coverage is referring to the availability of a large number of vehicles, which is required for the model. The system works with transfer hubs to consolidate passengers and then take these pooled passengers to a destination using one vehicle. The design strictly eliminates more than one transfer for any passenger. The pooled passengers are using vehicles with a reroutable and non-reroutable portion of their travel plan. The simulations performed in the research are strictly meant to show initial feasibility. This means that certain simplifying assumptions have been made and further they show the worst-case scenario. For their solution method they make use of the Nearest Neighbor Algorithm and more specific heuristics in order to create a good solution to their simulation model.

The research of Cortés et al. (2010) relaxes the coupling constraint in the PDP with time windows. The coupling constraint told the system that a passenger had to be always allocated to one vehicle from its pickup to its delivery location. This decoupling between passenger and vehicles allows for transfers of the passengers between different vehicles at specified fixed locations. They created an arc-based formulation for this formulation and provide an exact solution method by using a branch-and-cut algorithm which is based on Benders’ Decomposition (Benders, 1962). This branch-and-cut algorithm is the Combinatorial Benders Cuts introduced by Codato and Fischetti (2004). The objective function is to minimize the total ride time spent by the fleet of vehicles. In their research they explain how to model transfer nodes into mathematical formulations.

2.2.2 Cargo Hitching

On September 25th 2007 the European Commission made the following statement “Urban freight distribution could be better integrated within local policy-making and institutional settings. Public passenger transport is usually supervised by the competent administrative body while freight transport distribution is normally a task for the private sector. Local authorities need to consider all urban logistics related to passenger and freight transport together as a single logistics system” (European Commission, 2007).

Trentini and Malhéné (2010) were quick to react to this statement with a survey on these possibilities of sharing the road in urban cities. In their survey they made an overview of existing concepts and future concepts. In the article of the European Commission they state a specialist which came up with a design of a radical urban transport solution, called Freight*Bus (Frost, 2008), which he believes that can tackle
a lot of city logistic problems. For the city of London he designed a new bus concept which could be
used for both passengers as well as for packages. The article also mentions earlier ideas of sharing
different cargo and passengers in for example subways (Van Binsbergen & Visser, 1999). Trentini and
Malhéné also categorize the integrated use of busses as *shared buses* and the integrated use of taxis as
*ride sharing*. Furuhata et al. (2013) presented a classification to understand the key aspects of existing
ridesharing systems. Their objective is to present a framework that can help identify key challenges in
the widespread use of ridesharing and thus foster the development of effective formal ridesharing
mechanisms. Car sharing mods most of the time focus on the matching between parties. Agatz et al.
(2012) mention three different arrangements for ridesharing: single rider, single driver arrangement;
single driver, multiple rider arrangement; and single rider, multiple driver arrangement. In order to solve
the models they propose multiple solutions found in previous research.

A model which uses time windows, transshipments, scheduled lines and Cargo Hitching is created by
Ghilas et al. (2014), they created a mixed-integer formulation for the PDP with Scheduled Lines (PDP-
SL). In their paper the scheduled lines are bus lines with a fixed predetermined route and timetable.
According to this timetable the buses can only start their route at fixed departure times. In their model
the scheduled lines are replicated for every cargo request. This replication is performed to keep track of
the time synchronization constraints. The replication of scheduled lines has been proposed earlier in the
paper by Hall et al. (2009). The taxis in the model of Ghilas et al. (2014) are used for transporting cargo
as well as for passenger and therefore they have used different requirements for the packages and the
passengers. Further the passengers their tolerance towards additional travel time is considered as a
variable in the model. In this model only one scheduled line was used, their cargo requests were all size
of 1 and they used a standard Solomon benchmark dataset. This model is one of the most complete
formulations that exists for Cargo Hitching to the best of my knowledge. In order to solve their Mixed
Integer Linear Programming model they use a professional linear programming solver software called
CPLEX. To make the calculations more efficient they perform proposed cuts. The exact method is used
for a maximum of 10 requests. Further in this research they proposed and used a heuristic called the
Adaptive Large Neighborhood Search. With this heuristic they have solved cases up to 100 requests.

Li et al. (2014) considered conceptual and mathematical models in which people and parcels are handled
in an integrated way by the same taxi network. They propose two multi-commodity sharing models. The
Share-a-Ride Problem (SARP) and a reduced problem based on this SARP, the Freight Insertion
Problem (FIP). For these two problems they present a mixed integer linear programming formulation.
For the exact solving of these formulations they are using a linear programming solver software called
GUROBI. An important feature of the SARP is that there is the possibility to reject both people and
freight if the capacity is not sufficiently large. The objective function for SARP maximizes the total
profit which equals the income obtained from people and freight delivery minus the costs. For solving
the dynamic SARP they use general procedure algorithms. A neighborhood search is used to extend and
optimize the routes found. For initialization and optimization of a route, a greedy heuristic is created to find a feasible solution. The objective function of the FIP maximizes the profit from parcel delivery minus the cost related to the extra distance for parcel delivery and extra ride time for passengers. An important future implication is including environmental issues in the objective function. The FIP and SARP have been solved for 10 instances.

2.3 Conclusions

The pickup and delivery problem (PDP) is a generalized form of the vehicle routing problem (VRP). The first research on VRP dates back 50 years ago. This research was aimed at solving problems with a small number of requests and a small set of constraints. The first known successful attempt was made by Dantzig and Ramser in 1959. Their solution method was based upon linear programming. In those early years the computers were not highly advanced and thus there was a need to create heuristics which could solve problems without the help of computers. For a well-known example I refer to the Savings algorithm created by Clarke and Wright (1964). This is a heuristic which is able to solve the VRP with capacity constraints. Throughout the years researchers sought new ways for solving these VRP and also the PDP and their know variants.

So if we would look at those 50 years of developments then we see that the recent heuristics are highly accurate and are getting close to optimal solutions for PDPs. Also the exact algorithms are able to solve cases with more and more requests with Branch-and-Cut-and-Price algorithms which can solve up to 121 requests. The heuristic and exact algorithms are becoming more flexible and can solve more complex variants on the PDP.

If we would fast forward to now and I type ‘Pickup and Delivery Problem’ into the scholar search method by Google then it would return an approximation of 54,500 result hits. This is an incredible amount of research done on this topic. Many of the newer researches focus upon the more complex PDP models which resemble the practical distributions more closely. The recent technical improvements in the transportation field also allows the companies to use these methods. The technical improvement are for example advanced global positioning systems, radio frequency identification and parallel computing which could be taking into a vehicle.

2.4 Contribution

More research is focusing on concepts similar to Cargo Hitching. Many of these are a generalized version of the Cargo Hitching and almost no real world case studies. With this thesis I am going to try to fill this gap in the literature domain. Besides the literature, Cargo Hitching is already being deployed at some places in the world. Most of these integrations have the purpose for long-haul transportation, a big example is the air transportation and Hurtigruten which I covered in section 1.2.

I will investigate the opportunities for a private logistic service provider (LSP) to combine their own fleet with public transportation services in combination with taxi transportation services in order to serve
their cargo requests. The public transportation vehicles, for example busses, operates according to predetermined routes and timetables. Therefore when the LSP wants to cooperate with the public transportation company then its cargo has to be delivered towards a location on the bus route, called a transshipment hub. The cargo journey then continues with the use of a bus line towards another location on the bus route. A cargo request can then be collected by for example a taxi and its route can continue towards its final destination.

This case study will be modeled as an extension of the Pickup and Delivery Problem with Time Windows (PDPTW) in which integration of transfers and scheduled lines is permitted. A part of the request’s journey can thus be carried out with a public transportation vehicle. Both direct and indirect flows are considered. Direct flows represent the case where a request uses only door-to-door transportation (by using a pickup and delivery vehicle). Indirect flows represent the situation where a request is transported by a scheduled line as a part of the total journey. This specific transportation variation is denoted as a Pickup and Delivery Problem with Scheduled Lines (PDP-SL).

I consider two models in order to test the potential of the integration model. The first model will be the LSP in cooperation with the public and taxi transportation companies, the second model will not allow this cooperation and only permit direct delivery by the LSP. With the computational results I will be able to make conclusions about the potential benefits between the two systems. The PDP-SL is NP-hard since it is a more complex extension of the classical PDP-TW which is also NP-hard. For this reason, I have also developed a metaheuristics to obtain good-quality solutions within shorter computational times. The research performed by Ghilas et al. (2015) is fairly similar. They also assess the added value of the PDP-SL and create a heuristic method for creating quick and good solutions. The differences between Ghilas et al. (2015) their methods and mine will be discussed in Section 4.4 and Section 5.5.

Chapter 3
Lekkerland case study

3.1 A general description
For the case study I have access to the data of the company Lekkerland. Lekkerland is a multinational wholesaler and partner for all retail formats that give passengers convenient “on-the-go” consumption. The location Son has the biggest distribution center (DC) of Lekkerland in the Netherlands and contains a large fleet of heterogeneous vehicles. From this distribution center the Lekkerland fleet transports cargo to Noord-Brabant, Limburg, Zuid-Holland, Utrecht and Gelderland as well as cross-docking to distribution centers in Waddinxveen and Meppel.

3.2 Scope of case study
Integrating passengers and cargo can be realized with multiple different transportation modes. For the case study I broke down these possibilities before starting the case study. The first breakdown is into
two possible categories. The first possibility is letting passengers hitchhike with cargo transportation and the second possibility is letting cargo hitchhike with passenger transportation. Hitching means that a type of group, passengers or cargo, will make us of a transportation mode, which is already used, for a different type of group.

Choosing which option is better is based upon three differences between cargo and passenger transportation. First it is easier for passenger transportation to share their capacity with cargo than the other way around. When we want passengers to use the unused capacity of cargo transportation than a lot of modification have to be made to the transportation vehicles. These modification have to increase for example the safety and comfort. The modification of public transport is less extreme and requires creating cargo space. This means that the easiest option for increasing utilized capacity is the modification on the passenger transportation side. A second difference is that passenger transportation routes are more predictable then cargo transportation, for example all of the bus routes and departure times are fixed as well as a large portion of student transport taxi routes. The third difference is that cargo is less sensitive to changes in duration, departure time and other inconveniences. Based on these differences the decision has been made to choose the integration of cargo into the passenger transportation for the scope of the case study and the integration of passengers into cargo transportation is outside of the scope.

Almost 50% of ambient clients of Lekkerland Son are clustered in Noord-Brabant. The cargo sharing problem is a very complex problem with a great number of constraints and decision variables. Every day on an average of 358 locations has to be visited by Lekkerland in Noord-Brabant alone. This is already a very large sample size for the cargo sharing problem and therefore the scope of the thesis will be focused on Noord-Brabant or even more specific on Eindhoven and surroundings. The average number of unique clients which require delivery in a week for the city of Eindhoven and its surroundings can be found in Appendix 3.1. In this figure the number in the municipalities denote the number of unique clients which have to be visited in a day.

The next in depth specification is the choice of which passenger transportation modes are to be included; waterway, bus, subway, train, taxi or tram. Trains are an option because the DC in Son supplies many kiosks in train stations. The down side is that these municipalities where DC son is active do not have many train stations and therefore the potential gain is not very large for these regions. Next the subways and trams are also no option because they are not present in Noord-Brabant.

Road transport has more potential than water transport mainly because the locations of the distribution center in Son and the clients are not close to the shore or large water lines. What is left is the road transport and this includes the bus system and taxi operators, which have a great potential for integration between passengers and cargo. Many bus routes, especially on off-peak hours, are lower in utilization of capacity than desired. These buses still drive their routes according to a fixed time schedule to give
passengers the option to travel with public transport even on off-peak hours instead of getting a more expensive taxi. The routes and times are also fixed and this creates simplicity and transparency for passengers so they know exactly where they have to be at what time to catch a bus ride.

In the delivery system used by for example Connexxion, around 80% of the taxis in Eindhoven have fixed pickup/delivery locations and pre known pickup times. The taxis can be extremely useful as for example they can pick up cargo before they pick up passengers and drop the cargo off after or before the passengers are dropped off. When a large part of the route of the cargo and passengers is the same then the benefit of using one vehicle instead of two offers a good alternative. A portion of taxis is not constraint to scheduled routes or fixed time schedules. When they are not used, so in idle state, they are at a waiting location until a request comes in. During these idle times they can also be used to pick up cargo and drop it off at its delivery location. This way the capacity is utilized even during its previous ‘idle’ times.

The returning of empty containers or returned products is not in the scope of this research. This will require an even more complex model and is an interesting addition to the model. For the sake of simplicity the model will assume that if a location has a container from a previous delivery by a Lekkerland truck which must be taken back, that it will be stored at that location and will be taken back the next time a Lekkerland truck visits the location.

3.3 Research questions

In order to assess the potential for the Cargo Hitching in this case study I have developed three research questions. The result of the thesis will provide answers to this set of questions. The research questions regard the possible benefit from Cargo Hitching for Lekkerland, the public transport company and the society. Lekkerland could use less trucks and/or drive less miles for the delivery of their cargo. In return the public transportation company could benefit by receiving a financial compensation for letting their vehicles be used to transport cargo when there is unused capacity. The society could benefit from less trucks which are driving in the city center.

The research questions are summed up here:

- How much could Lekkerland benefit form Cargo Hitching?
- How much could the taxi and bus line company benefit from Cargo Hitching?
- How much could the society benefit from Cargo Hitching?

When I want to give a proper answer to these research questions I will have to be able to answer other questions first. These questions can be answered throughout the case study and I will further elaborate on them in chapter 6.1.

- Is it possible to capture the full case study situation in one linear mathematical model?
- Is it possible to find the optimal solution to this model using an exact algorithm?
- Is the exact algorithm fast enough for companies like Lekkerland and a taxi company?
• Which heuristics or algorithm can be used to create a good and fast solution to the mathematical model but not necessarily optimal?
• How good do these heuristics perform with respect to the optimal solution?
• Which transfer hub locations are suitable for the case study?
• What will be the impact when some parameters change? Think for example of the cost for using a scheduled line or the capacity of a taxi.

3.4 Problem description and assumptions

The case study requires that the model used for experimenting has to resemble the real world as close as possible in order to say something about the potential of the Cargo Hitching concept. On the other hand it needs assumptions in order to be translated into a mixed integer linear program. In this section I will elaborate on the choices which I have made regarding the real world.

For this case study I have access to data from a real world logistic service provider (LSP) and this data contains details about the depot, request locations and the vehicle fleet. Solving the case study requires information about the duration and distance between these locations in order to create routes. For the value of these lines I assumed to look at a fixed distance and duration between locations. The duration between locations can be used for creating a routing solution with correct departure and arrival times. The distance value is used to calculate the cost when a route has been established.

The case study involves two types of clients which require service, cargo and passengers. The cargo clients are the clients who have ordered cargo to be delivered to them by the LSP. The passenger clients are the clients of the taxi company which requested their service to be transported from their pickup location to their delivery location. The cargo delivery orders and the passenger transport clients are called requests. A cargo delivery order is called a cargo request and a passenger transport clients is called a passenger request. All of the requests have a pickup and a drop off location. For both of these locations the clients required hard time windows which means that that location must be visited by a vehicle between the time windows and cannot be visited any minute earlier or later. It takes a certain service time when a vehicle visits the pickup or drop off location. Two examples of service times are unloading of a truck or letting passengers out. The duration of this service time is assumed to be fixed and known beforehand. The delivery time window of a passenger is based on the pickup time plus the normal travel time from the pickup to the delivery location plus the service time and plus a certain slack duration. This slack duration is the time which a passenger can be delayed because the taxi services a cargo request between the pickup and delivery location of the passenger. An example is the drop off of a cargo request at its delivery location if this lies close on the route of the passenger request.

In this case study it is assumed that there are two types of vehicles which do not follow a scheduled planning. These vehicles are assumed to have a fixed variable cost per kilometer. The cost in the optimization function for these vehicles is the cost which the LSP has to pay when it makes use of the
vehicles. This cost per kilometer for the LSP does not always equal the prime cost of a vehicle. For example if the LSP uses a taxi to deliver their cargo requests then it has to pay a certain amount per kilometer to the taxi company. The real cost per kilometer for the taxi company can be lower than what the LSP has to pay. The difference between these prices is the additional income for the taxi company. The vehicles also have a fixed capacity for cargo and passengers. All vehicles are allocated to a depot which means that they can only depart and return from that depot. These depots have a certain time window in which the vehicles can leave and return. It is also assumed that the vehicles cannot drive longer than a certain duration due to law and company regulations. All the vehicles have a certain size and I assume that their sizes are based on their capacity. This means that vehicles can have different sizes within a vehicle type. Some streets are not big enough for certain sizes of vehicles and therefore these vehicles cannot make a delivery at those streets. Finally I have assumed that the duration between nodes also depends on the vehicle type. This means that the duration between locations is different per vehicle type.

For the routing of cargo clients it is possible to make use of both type of vehicles but also busses. These busses have different properties than vehicles. Every bus has a certain start location and end location and cannot deviate from their original route between these locations. This also means that the cargo clients have to be delivered and picked up by a different vehicle from these locations. These start and end locations are called transshipment hubs. Besides a delivery location of cargo, the transshipment hubs are the only locations where it is allowed for cargo to be dropped off by a vehicle. The cargo then has to travel with a bus or has to be picked up by a taxi. The busses can only leave their start location at their predetermined departure time. It is also assumed that these busses have a fixed capacity for cargo clients.

The case study will only look at LSP clients which have orders that do not require conditioned delivery environments as for example a cooled environment. It is too expensive to adjust the busses and taxis to cope with the cooled environment. An illustration of the current and the proposed delivery system is shown in Figure 4 and Figure 5.

![Figure 4 Current LSP delivery system](image1)

![Figure 5 Proposed Cargo Hitching delivery system](image2)
Chapter 4
Exact Formulations

4.1 Notation

In this section, I will present a mathematical model of the current delivery system and the proposed system where scheduled lines and transshipments are allowed. These models are used to create a routing solution with an exact method and a metaheuristic. A few objects have to be clarified beforehand:

Requests. In the system every client type is defined as a request type. The set $P$ contains both of these request types. Set $P^1$ contains the first request type which is cargo and the set $P^2$ contains the second request type which is the passengers of the taxi company. Each requests $r$ has an origin location $o_r$ and a destination location $d_r$. Associated with the requests is their cargo demand $\mu^1_r$ and passenger demand $u^2_r$. This does not mean that a request has both a demand for cargo and a demand for passengers. In this case study a request has cargo demand or passenger demand. The time window for the pickup location, $[o^1_{r,0}, o^1_{r,u}]$, and for the delivery location, $[d^1_{r,0}, d^1_{r,u}]$ are hard and cannot be exceeded. The delivery locations $d_r$ of the requests $r$ cannot be visited by all trucks but have a maximum allowed size $\xi_r$ for trucks which perform the final delivery.

Vehicles. In the case study there are two types of vehicles. The set $V$ contains both of these vehicles types. Set $V^1$ contains the first type of vehicle which are the logistic service provider (LSP) trucks. Set $V^2$ contains the second type of vehicle which are the taxis. The vehicles $v$ have cargo carrying capacity $\theta^1_v$ and a passenger carrying capacity of $\theta^2_v$. These two types of requests do not exclude each other and therefore can carry as well cargo and a passenger request at one. Furthermore every vehicle has a certain cost per kilometer $\sigma_v$ and fixed cost $\tau_v$. The LSP trucks are also slower than taxis and therefore have a parameter $\rho_v$ allocated to them which contains how much slower they are. All of the vehicles have a time window $[o^2_v, o^2_u]$ in which they are operational, a maximum driving duration $\phi_v$ and a linked depot node $\gamma_v$ where they start their route and also will have to return at the end of the route.

Scheduled lines. The algorithm can choose scheduled lines, which are bus lines, for parts of a cargo requests route. On these bus lines only busses are used for transportation. The original bus lines are bundled in the set $O$. These bus lines have a start location $i$, end location $j$ and a predetermined set of departure time slots $Q_{ij}^q$ for leaving the node $i$. The values in this set are integer and denote the number of the time slot and not the specific departure time in that time slot. The departure times are saved in $\xi_{ij}^q$. This parameter holds the time that the bus departures over line $(i,j)$ when it has to leave according to time slot $q$. Every bus on a scheduled line has its own capacity $k_{ij}$ for cargo requests which decreases when more public passengers are using the bus on that scheduled line. There is a certain cost $\varphi$ associated for every cargo demand size that uses the service of a bus line.
Transfer hubs. The start and end locations of the scheduled lines are defined as transshipment hubs. The transshipment hubs have a certain service time $\beta_t$ when visited. They can be used as short term warehouse solutions but are not meant for long time storage. The short term storage means that a vehicle can drop off cargo and can leave before the bus or taxi arrives to pick up the cargo.

Replication. In the case study one or more vehicles may visit these transfer hub and can drop off multiple requests. This makes the time synchronization more difficult. How can we make sure that a request leaves at the correct time while other requests at the same node have to leave at another time and possibly towards a different location. Another issue is at what time are the vehicles allowed to leave that node with or without cargo if there are multiple departure and delivery times for the cargo at that location. In order to simplify the synchronization and keep better track of the request flows, these transshipment hubs and the scheduled lines of the busses will have to be replicated as previous done by Hall et al. 2009. To be more specific the transshipment hubs and busses have to be replicated by the number of cargo requests in the used data. This will create a large number of new replicated transfer hubs and busses in the system and therefore each of these replications can be only used to serve one corresponding cargo request. In other words every cargo request has its own transshipment hubs and bus. If we summate the used capacity by cargo of these replicated busses then this total used capacity has to be less than the original bus capacity. These replicated busses are bundles in the set $F$ and the corresponding replicated transshipment hubs $t$ are bundled in the set $T$.

Now I define the network $G = (N, A)$, where the set of all nodes is represented by $N$ and all of the arcs in the system are represented by $A$. For this network I assume that every arc can only be visited by the same vehicle once. These arcs have two additional values. The two values of the arcs are based upon the travel time and the total distance. The real duration of arcs in minutes is denoted by $\alpha_{ij}$ and the real distance in kilometers is denoted by $\pi_{ij}$. The duration values of vehicles are corrected with a certain factor $\rho_v$ because trucks generally travel slower than taxis.

The set $N$ consists of all requests their pickup and delivery locations, the depots and the replicated transfer hubs. The nodes $i \in N$ have a certain service time $\beta_i$ when they are visited. The nodes also carry a value $\lambda_{ir}^i$ which denotes if they are a pickup node, intermediate node or delivery node for the request $r$. The parameter $M$ is called the big $M$. This parameter has the value of 1440 because there are 1440 minutes in a day. All of the parameters and sets are provided in Table 1 and 2.

<table>
<thead>
<tr>
<th>Sets</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P$</td>
<td>Set of all requests</td>
</tr>
<tr>
<td>$P^1$</td>
<td>Set of all cargo pickup nodes</td>
</tr>
<tr>
<td>$P^2$</td>
<td>Set of all passenger pickup nodes</td>
</tr>
<tr>
<td>$D$</td>
<td>Set of all delivery nodes</td>
</tr>
<tr>
<td>$D^1$</td>
<td>Set of all cargo delivery nodes</td>
</tr>
<tr>
<td>$D^2$</td>
<td>Set of all passenger delivery nodes</td>
</tr>
<tr>
<td>$W$</td>
<td>Set of depots in the system</td>
</tr>
</tbody>
</table>

Table 1 Overview of sets
The model uses multiple decision variables to create feasible routes. The vehicle are linked to arcs with the binary decision variable $x_{ij}^v$ which equals 1 if the vehicle $v$ travels over the arc $(i,j) \in A$. The
requests are linked to arcs with the binary variable \( y_{ij}^r \) which equals 1 if the request \( r \) travels over the arc \((i, j) \in A\). In the case of the replicated scheduled lines if the variable \( z_{ij}^{rq} \) equals 1 then the request \( r \) travels over the arc \((i, j) \in F\) which departs according to time slot \( q \in Q_{ij}^r\). The time synchronization for the vehicle \( v \in V\) is being tracked by the variable \( l_v^r \) which tells the system at what time the vehicle leaves the node \( i \in N\). Every vehicle has a variable \( e_v\) which is different from \( f_{ij}^r\) and tracks the return time to the depot. This is split up because vehicles have the same node for their start and end but only one \( f_{ij}^r\). The time synchronization for the request \( r \in P\) is being tracked by the decision variable \( f_{ij}^r\) which tells the system the time that the request is finished at node \( i \in N\). Finally the decision variable \( u_v\) tells the system if a vehicle \( v \in V\) is used or not.

### Table 3 Overview of decision variables

<table>
<thead>
<tr>
<th>Decision variables</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( e_v)</td>
<td>Continuous variable – Equals the returning time of vehicle ( v) at its starting depot node ( w_v), ( \forall v \in V)</td>
</tr>
<tr>
<td>( f_{ij}^r)</td>
<td>Continuous variable – Equals the finishing departure time at node ( i \in R^4) of vehicle ( v \in V) for request ( r \in P)</td>
</tr>
<tr>
<td>( l_v^r)</td>
<td>Continuous variable – Equals the leaving time at node ( i \in N) of vehicle ( v), ( \forall i \in r^4, v \in V)</td>
</tr>
<tr>
<td>( u_v)</td>
<td>Decision – Equals 1 if vehicle ( v) is used, ( \forall v \in V)</td>
</tr>
<tr>
<td>( x_{ij}^r)</td>
<td>Decision – Equals 1 if arc ((i,j)) is used by vehicle ( v), ( \forall (i,j) \in A^v, v \in V)</td>
</tr>
<tr>
<td>( y_{ij}^r)</td>
<td>Decision – Equals 1 if route from node ( i) to node ( j) is used by request ( r), ( \forall (i,j) \in A^v, r \in P)</td>
</tr>
<tr>
<td>( z_{ij}^{rq})</td>
<td>Decision – Equals 1 if scheduled line from node ( i) to node ( j) transports request ( r) at time slot ( q), ( \forall (i,j) \in A^v, r \in P^2, q \in Q)</td>
</tr>
</tbody>
</table>

### 4.2 Mathematical formulation

In this section I will present two different mathematical formulations. The first formulation is a close resemblance to the current situation at Lekkerland and the second formulation incorporates the possibility of Scheduled Lines and taxis.

#### 4.2.1 Current Lekkerland model

The Capacitated PDP with Time windows can be formulated as the following mixed-integer formulation. The objective function minimizes the total distance traveled of all vehicles multiplied by the cost of the vehicle plus the fixed cost when a vehicle is being used in the solution.

\[
\min_{\sigma_v} \sum_{v \in V} \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}^r + \sum_{v \in V} u_v \tau_v
\]

1. \[
\sum_{j \in N} x_{ij}^r = u_v \quad \forall v \in V
\]
2. \[
\sum_{j \in N} \sum_{w \in W} x_{wj}^r = u_v \quad \forall v \in V
\]
3. \[
\sum_{i \in N} x_{ij}^r = u_v \quad \forall v \in V
\]
4. \[
x_{ij}^r \leq u_v \quad \forall v \in V, i,j \in N
\]
5. \[
\sum_{i \in N} \sum_{j \in N} x_{ij}^r \leq 1 \quad \forall i \in N
\]
6. \[
\sum_{i \in N} \sum_{j \in N} x_{ij}^r \leq 1 \quad \forall i \in N
\]
7. \[
\sum_{i \in N} x_{ij}^r - \sum_{i \in N} x_{ji}^r = 0 \quad \forall j \in N, v \in V
\]
Textual summary. Constraints (2) assures that when a vehicle is used in the solution that it has to leave its depot. Constraints (3) assure that a vehicle cannot leave from a different depot. Constraints (4) assure that a vehicle must return to its own depot at the end of the route. Constraints (5) assure that a vehicle can only use arcs once. Constraints (6) and constraints (7) assure that every node can only be visited and departed by one vehicle or less. Constraints (8) assure that if a vehicle travels to a node that the same vehicle must leave that node. Constraints (9) assure that a vehicle can only leave a node after it has left its previous node plus the traveling time from the previous node to the current node plus the additional service time of the current node. Constraints (10) assure that a vehicle cannot return to its depot before it has left its previous node plus the traveling time. Constraints (11) and constraints (12) assure that every requests can only enter and leave a node once or less. Constraints (13) assure that a request can only go into a node if it is not the pickup node of that request and can only leave a node if
that node is not the delivery node of the request. Constraints (14) assure that a request cannot leave a node before it has left the previous node plus the traveling time and current service time. Constraints (15) assure that for every request first the pickup location must be visited than the delivery location. A vehicle will have to visit a node within the time windows which is assured in constraints (16) and (17). Constraints (18) assure that a vehicle must return to its depot within its time windows. Constraints (19) assure that the cargo capacity of a LSP vehicle is not violated. Constraints (20) and constraints (21) assure the correct synchronization for the departure time of a vehicle and request after the visited the same node. Constraints (22) and constraints (23) synchronizes the finishing time of a request and the departure time of a vehicle at the delivery location of the request. Constraints (24) and constraints (25) assure that a vehicle can only be used within its own time windows, outside of these time windows the vehicles are not operational. Constraint (26) enforces a maximum drive time for every vehicle. Constraints (27) ensure that a vehicle is not too big for the delivery at the cargo requests its delivery locations. The domains of the decision variables are defined in constraints (28-33).

4.2.2. Proposed Lekkerland model
The proposed model includes the possibilities of transshipments, integrated scheduled lines and taxis. In the following part I will only show constraints which had to be changed and the constraints which had to be added in order to create a correct solution to the case study PDP-SL.

The objective function is the first part which requires a change for the new model. The cost not only incurs when vehicle are being used but also when a cargo request uses a scheduled line. This adds another term to the objective function.

\[
\min \sigma_v \sum \sum \sum x_{ij}^v \pi_{ij} + \sum u_v r_v + \varphi \sum \sum \sum z_{ij}^p \mu_r^1
\]
\[ l_{i,v}^{p_1} \geq l_{i,v}^{p_2} \quad \forall v_1, v_2 \in V, r \in P \tag{15} \]
\[ l_i^r \geq v_i^r + \beta_i - M(1 - \sum_{j \in \mathcal{E}} x_{i,j}^v) \quad \forall i \in R^2, v \in V \tag{16} \]
\[ l_i^r \leq v_i^r + \beta_i + M(1 - \sum_{j \in \mathcal{E}} x_{i,j}^v) \quad \forall i \in R^2, v \in V \tag{17} \]
\[ e_v \leq v_{y_0}^v + M(1 - u_v) \quad \forall v \in V \tag{18} \]
\[ \sum_{r \in P} \mu^i_j \gamma_{ij}^r \leq \sum_{v \in V} x_{i,j}^v \theta_v^i \quad \forall i,j \in \mathcal{A}^x \tag{19} \]
\[ \sum_{r \in P} \mu^2_j \gamma_{ij}^r \leq \sum_{v \in V} x_{i,j}^v \theta_v^2 \quad \forall i,j \in \mathcal{A}^x \tag{20} \]
\[ f_i^r - l_i^r \geq -M(2 - \sum_{j \in \mathcal{E}} x_{i,j}^v) \quad \forall i \in R^2, r \in P, v \in V \tag{21} \]
\[ f_i^r - l_i^r \leq M(2 - \sum_{j \in \mathcal{E}} x_{i,j}^v) \quad \forall i \in R^2, r \in P, v \in V \tag{22} \]
\[ f_i^r - l_i^r \leq M(2 - \sum_{j \in \mathcal{E}} x_{i,j}^v) \quad \forall i \in R, r \in P, v \in V \tag{23} \]
\[ f_i^r - l_i^r \leq M(2 - \sum_{j \in \mathcal{E}} x_{i,j}^v) \quad \forall i \in R^2, r \in P, v \in V \tag{24} \]
\[ f_i^r - l_i^r \leq M(2 - \sum_{j \in \mathcal{E}} x_{i,j}^v) \quad \forall i \in R^2, r \in P, v \in V \tag{25} \]
\[ f_i^r - l_i^r \leq M(2 - \sum_{j \in \mathcal{E}} x_{i,j}^v) \quad \forall i \in R, r \in P, v \in V \tag{26} \]
\[ f_i^r - l_i^r \leq M(2 - \sum_{j \in \mathcal{E}} x_{i,j}^v) \quad \forall i \in R^2, r \in P, v \in V \tag{27} \]
\[ \sum_{q \in Q^i} z_{ij}^q = y_{ij} \quad \forall r \in P^1, (i,j) \in F^r \tag{28} \]
\[ \sum_{r \in P} \sum_{(a,b) \in F^i} \alpha_a z_{ab}^r \leq \kappa_{ij} \quad \forall (i,j) \in O, q \in Q^i \tag{29} \]
\[ f_i^r - e_{ij} \leq M(1 - z_{ij}^q) \quad \forall r \in P^1, (i,j) \in F^r, q \in Q^i \tag{30} \]
\[ f_i^r - e_{ij} \geq -M(1 - z_{ij}^q) \quad \forall r \in P^1, (i,j) \in F^r, q \in Q^i \tag{31} \]
\[ f_i^r - (\xi_{ij}^r + \alpha_{ij} + \beta_i) \geq -M(1 - z_{ij}^q) \quad \forall r \in P^1, (i,j) \in F^r, q \in Q^i \tag{32} \]
\[ x_{ij}^v \in \{0,1\} \quad \forall (i,j) \in \mathcal{A}^x, v \in V \tag{33} \]
\[ y_{ij}^v \in \{0,1\} \quad \forall (i,j) \in \mathcal{A}^y, v \in V \tag{34} \]
\[ l_i^v \in R^+ \quad \forall i \in \mathcal{N}, v \in V \tag{35} \]
\[ e_v \in R^+ \quad \forall v \in V \tag{36} \]
\[ f_i^r \in R^+ \quad \forall i \in R^2 \tag{37} \]
\[ u_v \in \{0,1\} \quad \forall v \in V \tag{38} \]
\[ z_{ij}^q \in \{0,1\} \quad \forall r \in P^1, (i,j) \in F^r, q \in Q^i \tag{39} \]

Decisions for new constraints. In the first model constraints (20) assures that a request cannot leave earlier than a vehicle if they both travel over the same arc and constraints (21) assures that a vehicle cannot leave earlier than a request if they both travel over an arc. In the new formulation all of the original transshipment nodes are replicated for every cargo request, but these replicated transshipment nodes are still part of the original transshipment node and the other replicated transshipment nodes. It is allowed for requests to travel within these connected replicated transshipment nodes without a vehicle. This makes the time synchronization a little more complex. When a vehicle delivers a cargo request at
a transshipment node in order to let the cargo request travel further with a scheduled line or travel to a connected replicated transshipment node than this vehicle may leave the transshipment node earlier than the request. In the old model vehicles and request always had to leave at the same time, in the new model this is not always the case. For this reason the constraints (24) is split up in two sections. The first section, constraints (36), does not include a transshipment node and is therefore equal to the former model. The second section, constraints (37), is added and only includes the start of an arc which is a transshipment node. This constraint secures that when a request and vehicle travel away from the transshipment node towards a pickup, delivery or depot node then the vehicle may not leave the node earlier than the request. However, if a request or vehicle does not travel away from the node or the case where a request travels towards another transshipment node (which is possible without vehicle) then the vehicle may leave earlier than the request. In the first model constraints (19) assured that the capacity of trucks was not violated. In the new model this constraint needs to be expanded to deal with passenger demand of requests too. For this reason I split the constraints into two parts. Constraints (19) deal with the LSP cargo requests capacity of vehicles and the constraints (35) ensure that also the passenger demand is not more than the passenger capacity of vehicles. Constraints (38) is added to assure that if a replicated scheduled line is being used for a request than that request also has to travel over the same arc. This automatically assures that a replicated scheduled line can only depart once. Constraints (39) are added to control for the capacity of the original scheduled lines. The scheduled lines are replicated for every cargo request and therefore the total capacity of an original scheduled line is the summation over these replicated scheduled lines. The constraint take this summation over all of the replicated scheduled lines for every departure time and assures that the total demand is lower than the original capacity. Constraints (40) and constraints (41) assure that if a replicated scheduled line is being chosen for a request at departure time slot \( q \) than that request also has to depart at the departure time in that time slot. Constraints (42) is occurs at the transshipment node which is at the end of a replicated scheduled line. A vehicle is not allowed to leave this transshipment node before the departure time in the time slot from the start of that replicated scheduled line plus the scheduled line travel time and the service time at the transshipment node.

The objective function minimizes the total distance traveled of all vehicles multiplied by the cost of the vehicle. Above this cost there is also an added fixed cost when a vehicle is being used in the solution. The last cost part is the price of using a scheduled lines for a cargo request.

### 4.3 Tightening of decision variables

In this model the decision variable \( y \) is created with a dimension of size \( N \times N \times P \). For example, if a model includes two depots, two cargo requests, two passenger requests and one schedules line then the size of \( N \) grows with every increase in one of these. This situation holds that \( N \) includes two depot nodes, four pickup nodes, four delivery nodes, and four transfer nodes which is 14 nodes. The dimension of variable \( y \) is for this situation 14 x 14 x 4 or 784 variables. Adding only one cargo request increases
the number of decision variables to 1620 and after adding another request 2904 variables. Every of these variables can have the value 0 or 1 which results in a possible $2^{2904}$ combinations.

For this reason it is beneficial to reduce the number of decision variables, like $y$, which the algorithm creates. A simple example of this reduction is when in the PSP-SL system it is not possible to pick up a new passenger request when one is already present in the taxi. Therefore no value of $y_{ij}^r$ can exist traveling from a passenger pickup node to another passenger pickup node or to the delivery node of another passenger. This tightening of the possible decision variables can decrease the complexity of the model significantly. This can also be done for the vehicle arc decision variable $x$.

I will first show why many vehicle travel arcs $x_{ij}^r$ do not have to be created in the model. In order to keep this as clear as possible I have categorized the arcs. There are seven categories of nodes at which an arc can start or end:

- **W**: Depots set
- **P1**: Cargo request pickup location set
- **P2**: Passenger request pickup nodes set
- **D1**: Cargo request delivery locations set
- **D2**: Passenger request delivery nodes set
- **T1**: Transshipment nodes at start of replicated scheduled lines set
- **T2**: Transshipment nodes at end of replicated scheduled lines set

This means that there are 49 possible categories of arcs which can be created in the system. The choice to create or not also depends on the type of vehicle and therefore we have 98 possible options. I will globally explain why I made the choice of creating the arc or not. To make the explanation a little bit more clear in the text I will substitute the set of all nodes by $N$, depots by $W$, cargo request pickup nodes by $P_1$, passenger request pickup nodes by $P_2$, cargo request delivery nodes by $D_1$, passenger request delivery locations by $D_2$, start of scheduled lines transshipment nodes as $T_1$, end of scheduled lines transshipment nodes by $T_2$, LSP trucks by $V_1$ and taxis by $V_2$.

**Depots (W).** A vehicle can only leave the depot to pick up a request. The pickup of a request can be at a node in $P_1$, $P_2$ or at $T_2$. A vehicle in $V_1$ is not allowed to pick up a passenger request and can only travel towards a node in $P_1$. The opposite is true for a vehicle in $V_2$ and therefore can only go to a node in $P_2$. The rest of the vehicle arcs will not be created.

**Cargo request pickup (P1).** Only a vehicle in $V_1$ can pick up a cargo request and therefore the vehicle arcs going from a node in $P_1$ to a node in $N$ will only be created for a vehicle in $V_1$. From nodes in $P_1$ vehicles can only travel towards another node in $P_1$, $D_1$ or $T_1$. The rest of the arcs are not created because vehicles in $V_1$ are not allowed to travel towards $P_2$ or $D_2$ and will not pickup requests at $T_2$. After picking up any request no vehicle can travel to its depot before dropping this request off.
Passenger request pickup (P2). Only vehicles in V2 can pick up a passenger request node and therefore the vehicle arcs going from a node in P2 to any other node in N will only be created for vehicle of type V2. From nodes in P2 a vehicle can only travel towards a node T2, D1 or D2. The node in D2 is a special case in which this node must be the delivery node of the current passenger request pickup because a vehicle in V1 can only carry 1 passenger request. The rest of the arcs are not created because a vehicle in V2 may not pick up a request at nodes in P1 or pick up a second passenger at a node in P2. It is also not possible to deliver cargo to a node in T1 or to a node in W.

Cargo request delivery (D1). These nodes can be visited by vehicles in V1 and V2. From a node in D1 it is possible to travel to a node in P1 but only for a vehicle in V1. It is also possible to travel to nodes in P2 and D2 but only if this is done by a vehicle in V2 because vehicles in V1 may not visit passenger request nodes. Arcs are also created which go to other nodes in D1, T2 and W for both vehicle types.

Passenger request delivery (D2). These nodes can only be visited by vehicles in V2. These vehicles can only visit nodes in P2, D1, T2 and W afterwards. P1 and T1 can never be visited by these vehicles and D2 is not an option because then it would carry multiple passenger requests.

Start of replicated scheduled line (T1). The nodes in T1 can only be visited by vehicles in V1. From these nodes in T1 only vehicle arcs are created towards cargo pickup or delivery nodes in P1 and D1 or towards a node in W. It is not possible to travel towards a node in T1 or T2 because requests can travel between transshipment nodes without a vehicle.

End of replicated scheduled line (T2). The nodes in T2 can only be visited by vehicles in V2. From the nodes in T2 only vehicle arcs are created towards passenger pickup or delivery in P2 and D2 or nodes in D1 and T2. Nodes in P1 and T1 are not created because they cannot be visited by vehicles in V2 and nodes in W are not created because the vehicle must first deliver the request it picked up at the node in T2.

The same has been done for the decision variable $y_{ij}^r$ which are the request travel arcs. A summation of the tightening of $x_{ij}^r$ and $y_{ij}^r$ is given in Appendix 4.1 and Appendix 4.2. Tightening is also used for the decision variable $z_{ij}^{rq}$. This variable tells the system if a scheduled line from node $i$ to node $j$ is used by request $r$ which departs at time slot $q$. The original scheduled lines are replicated for every cargo request which means that every replicated scheduled lines has its own associated request $r$. The algorithm only creates the arc for variable $z$ if the start transshipment node, end transshipment node and replicated scheduled line are associated with the same cargo request.

4.4 Comparison to Ghilas et al.
Ghilas et al. 2015 previously created a mathematical formulation for the PDP-SL. His formulation and constraints explanation can be found in Appendix 4.3 and Appendix 4.4. Our formulations closely resemble each other because they are based on the same routing problem with almost the same
constraints. Constraints which we have in common are for example cargo and passenger pickup and drop-off time windows, heterogeneous vehicles and multiple depots. There are however differences in the formulation. I will first discuss the differences between our objective functions and then the differences in our constraints.

**Objective function.** The first difference in the objective function is that different vehicle also have a different variable cost per kilometer in the objective function. The reasoning is that trucks have a higher cost than taxi for example. But even between trucks there can be a difference in cost per kilometer. A second difference is that my objective function looks at total kilometers drive instead of duration of a vehicle. The third constraint is that I also added a cost in the objective function which occurs when a vehicle is used in the routing solution. The reasoning is that it is beneficial for the LSP when they can reduce the number of vehicles in their fleet and also the environment benefits form less vehicles.

**Constraints.** A first difference in constraints is that I use two different capacity types, cargo and passenger. Ghilas et al. only uses the cargo carrying capacity for its Pickup and Delivery vehicles. A second difference is a result of the way I let cargo requests travel from a replicated transshipment hub to another replicated transshipment hub which is a replica of the same original transshipment hub. In Ghilas et al. a vehicle must drop off a cargo request at it associated replicated transshipment hub. In my formulation it is possible for a vehicle to drop off multiple cargo requests at a replicated transshipment hub and then a cargo request can travel to its own associated replicated transshipment hub without the vehicle. A third difference is that I have added maximum drive time and time windows for vehicles. The reason for vehicle time windows is that taxis are not always operational when the corresponding depot is. A fourth difference is in my formulation a cargo delivery location cannot be visited by all sizes of vehicles because of street impediments. Furthermore a few constraints are different for in attempt to decrease the CPU time.

**Chapter 5**

**PDP-SL Metaheuristic**

Solving the MILP formulation with a standard commercial solver is for many situations not a fast enough method. In order for companies to use the Cargo Hitching concept they need a relatively fast solving method. As explained in chapter 2 there are heuristic and metaheuristic algorithms which can create a good solution within an acceptable run time. For the pickup and delivery with scheduled lines (PDP-SL) there is not a great amount of research on this topic. Ghilas et al. (2015) has used an adaptive version of the Large Neighborhood Search (LNS) and had good success. The LNS is therefore a good option as improvement heuristic, but first a construction heuristics has to be used to create an initial solution.

For the construction heuristic I created a method which exists of three sequential phases. For the first two phases the algorithm has the option to select any LSP truck or taxi. In the first phase the algorithm
solves creates a feasible solution in which all of the cargo requests are picked up and delivered within their time windows. The second phase extends this solution and creates a new feasible solution in which also the passenger requests are being picked up and delivered according to their time windows. The third phase introduces the scheduled lines. This phase tries to combines all of the vehicles with the scheduled lines in order to create an initial solution for the PDP-SL in which all of the requests are picked up and delivered.

The fourth phase is the improvement heuristic which is based upon the large neighborhood search (LNS) method in combination with elements from the tabu search and some smaller heuristics. The input for this heuristic is the initial solution from the third phase. In the next section first I will present a higher level overview of the phases and this will be followed by a more detailed description of the different phases.

5.1 Algorithm phases

First phase. The basic idea is to first create a feasible solution for the LSP trucks and the cargo requests without using scheduled lines. The algorithm has two standard methods for the creation of a feasible solution. In the first method the algorithm uses the greedy heuristic for selecting the cargo requests. The greedy heuristic is an algorithm that makes the locally optimal choice at each stage. This means that every time it has to decide, that it will only look at the current solution and what will happen to the solution if it chooses an option. The greedy selection criteria on which this choice is based is the distance between the delivery location of the request and the delivery location of last added request or the depot if the route is still empty. The algorithm continues this method until a feasible routing solution is created for all cargo request. The second method uses the solution of an exact method with the old model as input. The algorithm can choose further on which initial solution provided the best results.

Second phase. The second phase of the heuristic creates a feasible solution for the taxis and the passenger requests. For the creation of this solution the greedy heuristic is again used. The earliest customer is selected with the earliest possible taxi. Continuing with this vehicle the algorithm searches the customer which has be picked up earliest after the drop-off of the last passenger which is also close by. When the taxi has no more possibilities than the next taxi is chosen.

Third phase. The next phase of the heuristic tries to insert scheduled lines where it combines the vehicle routes in the previous phases. It chooses a cargo request with high additional cost in the solution and with its drop-off location close to a transfer hub. When this cargo request is chosen the algorithm chooses the most suitable scheduled line and departure time. The next step is to find a taxi which has the least added distance when it picks up the cargo request from the transfer hub and drops it off at the destination location. Now the algorithm checks if all constraints are met and if the cost had decreased. If it is decreased then it accepts this new solution, if not decreased or if the constraints are not met than this solution will be put on the tabu list.
**Fourth phase.** The fourth phase involves the improvement heuristics. There are five different methods used in this phase.

5.1 *First phase: Initialization truck routes*

The code snippet with comments of this phase can be found in Appendix 5.1. This phase starts with creating a list $L$ of all the requests left in the system and two empty lists $T^r$ and $T^v$ which contain requests and vehicle if they are on the tabu list. The algorithm will then create two empty decision variables, $newX$ and $bestX$. The decision variables $newX$ and $bestX$ correspond with $x_{ij}^v$ in the mathematical formulation. The system always keeps track of the last location in the route or if the route is empty than it remembers the depot. The value of this last location or depot is stored in the variable $i$.

Now the algorithm will select a vehicle with the largest capacity and saves it in the variable $v$. The reason it chooses the vehicle with the largest capacity is to try to and use as few trucks as possible. Therefore the goal is to maximize the use of this truck before selecting another. In the improvement phase the algorithm tries to use smaller trucks if the capacity is not fully used. Now the request with the delivery location closest to the depot and which is not on the tabu list is selected and saved in the variable $d$ and its pickup location is saved in variable $p$. The algorithm chooses based on the delivery location and not pick up location because in this case study the pickup location is located at the same location as the depot and would therefore always equal zero. The algorithm will insert the selected cargo request in the decision variable $newX$. The insert function tries to insert this request at all places in the current route if there are already requests in the $newX$ solution. If there are no requests in the route yet then it will just be placed between the depots. For every insert attempt the algorithm selects the place in the route with the lowest additional distance compared to the old solution. If all the constraints are met then this solution will be saved in the decision variable $bestX$ and the request will be removed from the list $L$. If the constraints are not met then this route will be put on the tabu list and the algorithm chooses a different request. If a request is on the tabu list $T^r$ then it will not be selected by the algorithm until the algorithm has emptied the tabu list. It empties the tabu list if a new vehicle is selected and therefore the request could be an option again. When there are no more possibilities left for this vehicle then the algorithm puts the current vehicle in the list $T^v$ and selects a different vehicle which is not in list $T^v$. This algorithm will continue this process until the list $L$ is empty or if no more possibilities exist.

5.2 *Second phase: Initialization taxi routes*

The code snippet with comments of this phase can be found in Appendix 5.2. This phase starts again with the initialization of the list $L$ which now holds all of the passenger requests and initializes the list $V$ which holds all the available taxis. The algorithm also initializes the empty tabu list $T^r$ and the cluster list $C$. The algorithm creates the decision variable $newX$ which duplicates the values from the variable $bestX$ from the first phase. It also creates the decision variable $bestX$. 
The algorithm will now create clusters of the passenger requests in the dataset. The algorithm decides in which cluster a request belongs based on the delivery location of the request. For every cluster the algorithm will choose the taxi depot which is closest to the center of the cluster and saves this in the variable \( w \).

The algorithm will select the vehicle with the earliest available time which is also allocated to current depot in variable \( w \) and saves this vehicle in variable \( v \). When the taxi is selected then the algorithm finds the passenger request which has the earliest pickup time and is not in the tabu list \( T^r \). The delivery location of this request is saved in variable \( d \) and the pickup location of this request is saved in variable \( p \). This request will be placed at the end of the taxi route in the variable \( newX \). The algorithm will then check for \( newX \) if all constraints are not violated and if not violated then duplicates the variable \( newX \) into the variable \( bestX \). If any constraint is violated then the current decision variable \( newX \) is placed in the tabu list \( T^r \) and \( newX \) duplicates the decision variable \( bestX \). The decision variable \( newX \) duplicates \( bestX \) because \( bestX \) is the current best solution which does not violate any constraint. The algorithm repeats these steps until no passenger requests can be inserted in the current taxi. When there are no possibilities left for this taxi but there are still passengers left in the list \( L \) then another taxi is chosen. If no passengers are left in the list \( L \) then this phase ends and phase 3 starts. If there are no vehicles left in \( V \) but there are requests left in list \( L \) then the algorithm will end and did not find a feasible solution.

### 5.3 Third phase: Insert Scheduled Lines

The code snippet with comments for this phase can be found in Appendix 5.3. The third phase of the heuristic is the first attempt of the algorithm to insert scheduled lines. It works as a hill climbing procedure with elements of the tabu search. First the algorithm creates an empty list in which the tabu solutions can be placed. Further the variable \( oldCost \) will receives the value of the cost of the current solution. This current solution is the result of the first two phases. The two decision variables \( newX \) and \( bestX \) will duplicate the decision variable \( initX \) which is the result of the previous two phases. The new binary decision variable \( newZ \) will hold the value 1 if a replicated scheduled line is used in the solution for a certain request and departure time and holds the value 0 if the scheduled line is not used. The variable \( newZ \) will hold the value 0 for all replicated scheduled lines at the beginning of this phase. This binary decision variable is the same variable as \( z_{ij}^{rq} \) in the mathematical formulation.

The algorithm will search the cargo request which has creates the greatest cost saving when removed from a route. The saving is calculated by looking at the cost difference between the current solution and the solution if this request is removed. The pickup location of this request is saved in the variable \( r \). The vehicle which served the request will be saved in the variable \( v \). Based on the location of the request its delivery location and its time window the algorithm will choose the most suitable replicated scheduled line. The most suitable replicated scheduled line is the line with its end transshipment node close to the
delivery location and the start transshipment node close to the pickup location. The next step is to find a suitable departure time and then insert the replicated scheduled line into the current solution. The insert of a scheduled line is done by setting the value of that line to 1 in the variable \(newZ\). Also the start transshipment node is added to route of vehicle \(v\) at the location which creates the least additional cost. After the scheduled lines is inserted, the request is not yet at its delivery location because a taxi has to pick it up from the end transshipment location and deliver the request to its delivery location. If the algorithm fails to find a feasible insertion then it will try a different departure time for the selected scheduled line and add the previous attempted solution to the tabu list. If the insertion is not successful and no more departure options are possible then the algorithm will select a different scheduled line. If no possibility is found for the request then it will be added to the tabu list \(T^r\). If the algorithm is successful in creating a feasible solution with a scheduled line then it will check if the cost of the solution has decreased. If it hasn’t decreased then the request will be added to the tabu list \(T^r\) and a next request is chosen. If it has improved than the new solution will be saved in \(bestX\) and a new request is chosen. This process will continue until all requests are checked and no more improvement are found. The output of this phase is a feasible solution with the option of scheduled lines and transshipments.

5.4 Fourth phase: Improvement

The final phase of the heuristic is a combination of improvement heuristics. In total there are five different methods which will be performed sequentially. The first method tries to insert two scheduled lines at once for a certain cluster of cargo requests. The second method tries to add one or multiple scheduled line for cargo requests. It also tries to add another scheduled line to cargo requests which already use a scheduled line. The third method tries to integrate a scheduled line for multiple batches of requests and checks for improvement afterwards. This method tries different sizes for the batches. The fourth method tries to swap arcs in the vehicle routes, inter-route 2-opt, and tries to swap requests between vehicles, intra-route 2-opt. The final method tries to use smaller trucks if the capacity of a truck is not fully used.

5.4.1 Method 1 Insert cluster of request with (multiple) scheduled lines

The code snippet with comments for this method can be found in Appendix 5.4. This method is created in order to increase the number of scheduled lines used by clusters or larger requests. Clusters of cargo request which lie in other cities than the depot have a smaller chance to adapt a scheduled line than individual cargo request outliers. The reason lies in the Hill Climbing principle. As mentioned before, the Hill Climbing method searches the direct neighborhood which can be reached by changing a small operator. Imagine the situation where a truck has to visit Helmond for five cargo requests which lie relatively close to each other. The third phase of my PDP-SL heuristic tries to remove one of these requests from the truck route and chooses to let it travel via a scheduled line towards Helmond. This request has to be picked up by a taxi and afterwards the algorithm will check if the solution has improved. The problem with this example is that the LSP truck still has to travel almost the same route
because of the other four cargo requests which are left in the truck route. There is a high possibility that the removing of the delivery of the cargo request from the truck will not improve the solution and therefore this integration of a scheduled line is not accepted by the algorithm. The chances of acceptance is higher when we would search in a larger neighborhood which would let all of the five requests travel over a scheduled line before checking if the solution has improved. The reason is that the LSP truck now does not has to travel towards Helmond at all. Another example is a cargo request of five trolleys. If the capacity of taxis is 1 trolley then this cargo request is too large for taxis and therefore will be split up into five different cargo requests with the same delivery location. Removing one of these smaller requests and letting it travel over a scheduled lines will not decrease the total distance of the LSP trucks because it still has to go to the delivery location of the remaining four sub cargo requests.

The Large Neighborhood Search (LNS) method can create a better solution than the initial Hill Climbing heuristic. It has been shown by Ghilas et al. (2015) that this is a very good performing heuristic for this kind of problems. This first improvement method is therefore based upon the LNS. First the algorithm creates clusters of the cargo requests based on their delivery locations. The location of this cluster is used to choose the closest taxi depot location and the algorithm creates a list of the taxis associated with the depot. In the next step the algorithm removes all of the delivery locations of the requests in the cluster from their current routes. The algorithm will try to visit these delivery locations with a different vehicle after using a replicated scheduled lines. Now the algorithm tries to insert a scheduled line for every request in the cluster without violating time or capacity constraints. If there isn’t a direct scheduled line but only an indirect scheduled line then it will insert multiple scheduled lines per request in order to create this indirect scheduled line. If the delivery locations is for example in Helmond then first the scheduled line from Son towards Eindhoven central station and then form Eindhoven Central station to Helmond Central can be used. After the algorithm has tried to insert a scheduled line for all of the requests in the cluster then it will check if the solution has improved and accepted if so.

5.4.2 Method 2 Insert one or multiple scheduled line at once
The code snippet with comments for this method can be found in Appendix 5.5. This method is created to insert a scheduled line if a request already uses a scheduled line or could benefit from two scheduled lines or more. The reason for this method is that the initial heuristic mainly focusses on inserting one scheduled line. If there is a cargo request outlier which has a big benefit of using two scheduled lines then the construction heuristic still will not find it. There is a chance that the first improvement method in chapter 5.4.1 will find a scheduled line for this outlier but there is an exception where the outlier lies close enough to be part of a cluster but that the cluster as a whole is not an improvement for scheduled lines. This can be due to for example capacity problems. This second improvement method tries to insert one or more schedules lines for a request without intermediate testing if the cost has decreased. It can be so that the request already uses one scheduled line but will benefit from another, then this method will try one another line.
The first step is to create a list of requests which are an outlier. Outliers are requests with a distance of their delivery location the closest other delivery location of more than a certain kilometers. For every request in this list the algorithm will delete the delivery node from its current route. This leaves two options: the last known location of the request is the pickup location or it already was using a scheduled line so the last known location is the end of a scheduled line. The algorithm now searches for a scheduled line which lies closest to the request’s delivery location and select the appropriate replicated scheduled line for the request. The algorithm checks if an additional scheduled line is needed. If these scheduled lines are inserted then the algorithm will insert a taxi for the last mile and check if no constraint is violated. If the constraints are not violated then the algorithm will check if the cost is improved and if so then the new solution will be accepted. The request will now be removed from the request list and the algorithm continues until the list of requests is empty.

5.4.3 Method 3 Insert random clusters with scheduled lines
The code snippet with comments for this method can be found in Appendix 5.6. The first improvement method looked only for specific clusters in the requests. There is also the possibility to test all of the possible request batches without looking at their location. The advantage of this way of grouping requests is that the algorithm tests all possibilities instead of a select group as in the first improvement method. On the other hand, the advantage of select clustering is that it is generally faster as it tests fewer batches. Because select clustering is faster the algorithm can test larger clusters without comprising for too much CPU time.

This third improvement method uses the way of testing all possibilities for certain batch sizes. This method is comprised of three parts which differ in batch size. The first part tests batch sizes of two requests, the second part tests batch sizes of three requests and the third part tests batch sizes of four requests. All three parts use the same overall technique.

The algorithm starts with the creation of a list with cargo requests which are a direct delivery by the LSP, so they do not use scheduled lines. The next step is to sort this list from most expensive delivery to least expensive delivery. The most expensive delivery is the cargo request with the highest saving if deleted from the route. Then the algorithm will make sure that every combination is tested. It selects all combinations with allocating a binary variable to all of the requests. If the binary variable equals 1 then the request will be tested in the batch. For example if there are four requests in the list then there will be four binary variables which start as 0. If the batch size is two then the first combination will be with the first two variables equal to 1, remember that these two resulted in the highest savings. The requests which correspond with the 1 value will now be tested in the batch run. If this batch does not deliver improvement then the next combination will be created until there is an improvement. Table 4 shows the possible combination when using 4 requests.
A batch run starts with deleting the delivery location in the current solution for every request in the batch. Then it repeats the following process for every request in the batch. First the algorithm finds the replicated scheduled line which is closest to the delivery location of the request. It selects a suitable departure time and tries to insert the replicated scheduled line and the taxi pickup and delivery. If this is possible according to all constraints then it insert the LSP truck drop-off at the correct transshipment node. When this process has been done for all of the requests in the batch and no constraints are violated then the old and new cost will be compared. If the new cost is lower than the old cost then the new solution is accepted and the requests are removed from the list if not then the next combination will be tested. This process will continue until no requests are left or if the last combination has been checked and no improvement has been found. In the code snippet I will only show 1 out of the 3 parts of this method as the other two are almost identical.

5.4.4 Improvement method 4 Inter and Intra-route 2-opt
The fourth improvement method tries to swap arcs by performing 2-opt. This method again exists of multiple phases. The first phase performs the inter-route 2-opt. Inter-route 2-opt deletes two arcs in a route and switches them. After the switch has been performed the constraints will be tested and if they are not violated and the cost of the route is reduced then the new route solution will be accepted. This will be performed for all possible combinations in all vehicle routes. The second phase is the intra-route 2-opt and is very similar to the inter-route 2-opt. The main difference is that still two arcs are selected but these are not in the same route. Intra-route swapping of arcs is only within trucks.

5.4.5 Choosing the smallest vehicles
Improvement method 5 tries to use smaller vehicles when the LSP trucks are not fully loaded. The bigger the trucks the more they cost per distance unit. For this reason the algorithm will look for trucks which capacity is not fully used. For these routes it will check if a smaller vehicle is available and switch them. After switching the constraints are once again checked because not all trucks drive as fast. Probably the smaller trucks will be faster, but just to be sure they are checked.

5.5 Comparison to Ghilas et al.
Ghilas et al. (2015) also created a heuristic algorithm for the PDP-SL. They created an extension of the Large Neighborhood Search. This extension uses multiple large neighborhoods by applying several removal and insertion operators to a certain solution. For its initialization phase they used the greedy algorithm.
Ghilas and my initialization phase are both based on the greedy heuristic. A difference between our models is that I have two different request types for which a routing solution has to be created, namely cargo and passengers, and Ghilas is only concerned with the routing of cargo requests for this part. My heuristic initially uses LSP trucks for the cargo requests and uses taxis for the passenger requests. The second part of my initialization phase is the insertion of scheduled lines. Ghilas inserts the scheduled lines after the initialization phase. Another difference is that my heuristic splits up cargo requests which are too large for a taxi. In Ghilas every cargo request is smaller than the total cargo capacity of a vehicle and therefore this additional step was not required.

The biggest difference between our heuristic algorithms lies in the improvement phase. My heuristic uses pre-known information to improve the current solution, for example in the solution after the initialization phase all of the pickup locations are located at the depot of the LSP and are being picked up by a LSP truck. Ghilas uses several removal and insertion operators but my heuristic focusses on only on two type of removal and insertion operator. The first is the cluster removal and insertion method described in section 5.4.1 and the second is the batch removal and insertion method described in section 5.4.3. Besides the LNS elements I have used in the heuristic I also created a method which tries to insert more than one scheduled line at once for one request, this method is described in 5.4.2. The reason for this method is that a solution can include the use of multiple sequential scheduled lines for one request in my formulation. Another addition which my heuristic contains, which Ghilas its heuristic does not, is that it tries to swap all request from one truck to an empty truck which has a lower cost per kilometer.

Chapter 6
Computational results

The following chapter presents the results of numerical experiments that are designed to indicate the potential benefit for a LSP when integrating public scheduled lines. The heuristic described in chapter 5 is used to address the research questions from Chapter 2.2:

- How much could Lekkerland benefit form Cargo Hitching?
- How much could the taxi and bus company benefit from Cargo Hitching?
- How much could the society benefit from Cargo Hitching?

6.1 Experimental Design

In this part I will elaborate on the experiments that I have conducted to evaluate the case study. This evaluation is done by performing multiple experiments with different instances and parameter values. The different instances mean that I used different cargo and passenger requests which had to be served by different vehicles in the set. A parameter value is a value that has been set in the beginning of an experiment and does not change throughout the experiment. The parameter values hold the cost for the vehicles and scheduled lines for the experiments.
The cargo requests are based on real world data from the LSP Lekkerland. The passenger requests are based upon randomly generated pickup and delivery locations. These locations are extracted with a normal distributions which has its mean in the city center of Eindhoven and Helmond and they have a minimum allowed distance between the pickup and delivery location. The pickup times are generated by two normal distributions with their means in the rush hours. I generated 250 pickup time with their mean in the early rush hour and 250 pickup times with their mean in the afternoon rush hour. From these 500 passengers I have randomly selected passengers to be included in the dataset. The initial idea was to use the data for the company Connexxion but in Eindhoven they mainly focus on school transportation. These taxis have the same cargo request capacity as normal taxis. The problem is that these school taxis cause additional complexity for the exact method which results in long CPU time. The cause for this increased CPU time is that one school taxi carries multiple passenger request at once. It can transport for example 10 school students which are all separate requests in the model. While traveling with these 10 requests it can only hold up for example 1 trolley of cargo request at most. In order to deliver one cargo request the algorithm will use ten additional passenger requests instead of one as in the public taxi case. For this reason a taxi which can only hold only one passenger request at a time is more promising as vehicle in the case study.

If the volume of a cargo request is larger than the capacity of a taxi then it is not possible for a taxi to pick up the cargo request at the end of a scheduled line. For these requests no scheduled lines will be used. The algorithm used in this case study will split these cargo request into pieces smaller or equal to the taxi capacity. For example a cargo request of 10 trolleys is split up into 10 different requests of 1 trolley. This way it is always possible for a taxi with capacity of 1 trolley to pick up the cargo requests.

Earlier I mentioned to use multiple instances to assess the change in performance of the algorithm and to look at the potential of the Cargo Hitching system. For every instance the following parameters were also changed in order to assess the sensitivity and under which circumstances the concept works best.

- Kilometer cost of using cheapest trucks - €0.80, €0.90
- Kilometer cost for the LSP when using a taxis - €0.125, €0.30, €0.40
- Schedules lines cost for the LSP when using it for 1 trolley - €0.00, €0.50, €1.00, €2.00, €3.00

The algorithm is implemented in java and run on a laptop with a 3.0 GHZ dual core i7 Intel chip and 8 GB of RAM.

### 6.2 Experimental Results

In this section I will assess the benefit for the stakeholders based on the results of the experiments. This section is set up in four parts. The last three sections describe the benefits for the three different stakeholders which are the LSP, the taxi/bus company and the society.
6.2.1 General experiment setup

**Dataset specification.** For the general analysis I am going to highlight the salient points in the data sets and solution. In total I will use four different datasets for the experiments. These datasets contain information about the cargo requests which the LSP delivered for a Tuesday and a Wednesday in week 28 of 2014 and two different sets of generated taxi passengers. I have chosen these two days because they had the most unconditioned cargo requests. It was more interesting to experiment with more requests. The datasets consist of two parts. The first part contains information about the cargo requests and LSP truck vehicles extracted from the LSP database. The second part consists of the simulated data about the passenger requests, taxis and real world scheduled lines. All of the datasets assume that scheduled lines have a capacity of ten trolleys and all passenger requests have a maximum slack of ten minutes. This means that a passenger request can be delivered ten minutes later than its original delivery time. In the current real world this is already incorporated in the contract between the taxi company and the passenger.

**Tuesday.** This day the LSP had to visit ten unconditioned cargo request delivery locations to Eindhoven and 2 cargo requests in Helmond. In this data set I only discuss unconditioned cargo because it will be very expensive to create a cooled environment in busses and taxis as mentioned in the research description. Nine out of the ten cargo request towards Eindhoven had a demand size of 1 trolley or smaller. The taxi cargo capacity was set to 1 trolley and therefore larger cargo request demand sizes have been split into multiple cargo requests with smaller demand. One request towards Eindhoven was bigger with a demand size of 1.13 trolleys which therefore has been split into two cargo requests. One of the two cargo requests with their delivery location in Helmond its cargo demand size was bigger than one trolley, 2.24 trolleys to be exact. This request has been split up into three different requests by the algorithm and therefore there are four cargo requests with their delivery location in Helmond. In total there are now fifteen cargo requests in this dataset. The LSP has two different type of trucks suitable for the model. The first had capacity of 40 trolleys and the second had the capacity of 48 trolleys. The more capacity the truck contains the more expensive they are for the LSP per kilometer.

The Tuesday cargo requests was used in combination with two taxi passenger datasets. The first passenger dataset contains in total 45 passenger requests and the second contains 15 passenger requests. In order to serve these passenger requests there are five available taxis in Eindhoven and five available taxis in Helmond. The Eindhoven taxis have their depot in the center of Eindhoven and the Helmond taxis have their depot in the center of Helmond. The locations of these depots are based upon real world taxi companies their depot location.

This dataset also includes two schedules line and three associated transshipment hubs. The first scheduled line starts at the transshipment hub in Son and ends at the transshipment hub in Eindhoven central station. The second scheduled lines starts in Eindhoven central station and ends in Helmond station. The start and end locations of the scheduled lines had the requirement that they must be located
at real world bus stops. A second requirement was that these real world bus stops should lie at the end of a bus line. The reasoning for this is that these locations have the most potential for the required space and time. The space and time is needed to load and unload cargo requests. These locations have the most time potential because bus companies schedule a few minutes stop here in order to cope with deviating travel times. Another reason is that this is the only location on the route where bus drivers are allowed to step out of the bus.

With these requirements in mind I have chosen the start of the first scheduled line to be a bus stop as close as possible to the depot of the LSP. This choice of location for the start transshipment had the greatest potential to minimize the total distance of the truck routes because the truck didn’t had to deviate much from its original route when delivering cargo towards the scheduled line. The reason for the end of the first scheduled line is based again on travel distance minimization. The distance towards the delivery locations of cargo in Eindhoven is minimized at the station of Eindhoven. The start of the second scheduled line is chosen to be at the end of the first scheduled line. This eliminates the necessity for a vehicle to ship cargo from the first to the second scheduled line. The end of the second scheduled line follows the same reasoning as the end of the first line. It minimizes the distance between the delivery locations of cargo and passengers from Helmond.

**Wednesday.** This day the LSP had to deliver ten unconditioned cargo requests to delivery locations in Eindhoven and ten unconditioned cargo requests to delivery locations in Helmond. Multiple cargo requests had a demand size of more than one trolley. The algorithm had split these 20 cargo requests into 29 smaller cargo requests. The algorithm had again the option of three different trucks which had a different cargo capacity and cost function. The dataset included the same passenger requests, taxis and scheduled lines as in the Tuesday dataset. The reasoning for the locations of these passengers, taxis and scheduled lines is the same as in the Tuesday dataset.

**Parameter configuration.** The datasets were used as input for the experiments with eight different parameter settings. I chose to set the truck cost at €0.8 per km for a vehicle of cargo capacity of 40 based on research on average cost of a truck when delivering in urban areas (Cost of Heavy Vehicle Use in Canada, 2015). A vehicle of cargo capacity of 48 is €0.85 per km. I also included the parameter setting of €0.9 per truck km of cargo capacity of 40 trolleys and €0.95 per km for a truck of 48 trolley cargo capacity to assess what happens if this cost is increased. The price for the LSP to use a taxi has been set to €0.3 and €0.4 per km. This price is based on the additional driven kilometers by the taxis when they also have to service cargo. The cost for the taxi company is lower than this price for the LSP. The taxi drivers are already on duty whether they serve cargo requests or not and I assumed that there is also no increase in insurance and tax. For this reason I base the cost of a taxi per kilometer for the taxi company only on the variable cost. This cost excludes the insurance, driver cost and tax. A taxi drives an average of 4000 km per month (Cabture.nl, 2015) and most taxis drive on Diesel which have an average of 1 liter on 20 kilometers (GemiddeldGezien.nl, 2015). Most taxis have to accelerate and decelerate more
than normal drivers as they mostly drive in city centers. Therefore I set the average Diesel consumption of a taxi on 1 liter for 18 kilometers. The current Diesel price per liter is €1.347 (Dieselprijs.eu, 2015). The cost for Diesel is therefore $4000 \cdot \left( \frac{1}{18} \frac{1}{km} \right) \cdot \frac{1.347€}{l} = €299.33$ per month. The variable cost also includes maintenance and depreciation cost which I have estimated on €200 per month. This totals €499.33 of variable cost per month spread over 4000 km which is a total variable cost of €0.125 per kilometer. For this model is assumed that the additional variable cost price is around €0.125 per kilometer for the taxis. The cost for the LSP to use a scheduled line has to be estimated as this is not yet an option in the real world. The bus line company has minor additional variable cost when he allows cargo when there is capacity left. Therefore the bus line company will base the price it asks the LSP on the amount which maximizes his additional income. The experiment therefore parameter settings of €0, €0.5, €1, €2 and €3 per cargo trolley. An overview of the heuristic algorithm results is presented in Appendix 6.1, 6.2, 6.3 and 6.4.

A graphical illustration of the Wednesday dataset its locations with the 45 taxi passengers is presented in Figure 6. In this figure the blue triangle is the LSP depot, the big blue pin points are the cargo requests delivery location, the smaller brown pin points are the taxi pickup locations, the golden pin points are the taxi drop-off locations, the green pin points are the transshipment hubs and the blue line are the scheduled bus lines. Figure 7 shows the optimal solution for the current LSP delivery system. The red line is the LSP truck routing solution. Figure 8 is the same optimal solution for the LSP truck but also includes the routing solutions of one of the taxis, this taxi route is shown as the brown line. This taxi route does not serve any cargo requests yet. Figure 9 presents the heuristic solution for the delivery system which includes the option for scheduled lines and taxis to serve cargo requests. You can see that the red line (LSP truck) is indeed shorter but the brown line (taxi) is longer. What you cannot see in this figure is the use of the scheduled line. The numerical results show that 38 blue line (bus line) are used to transport cargo requests.
6.2.3 Lekkerland benefit from Cargo Hitching
In order to assess the benefit for the LSP Lekkerland I will mainly look at the total cost picture for the LSP. The exact method and the heuristic method their objective function was to reduce the total cost for the entire system including LSP truck, taxi cost and scheduled lines cost. The solution of the exact method and the heuristic method are used to compare the costs for Lekkerland in the different cases. The solution of the method include routing for the trucks and the taxis. It is easier to assess the cost for the LSP as a result of the truck routes because this is paid only by the LSP. However the taxi cost requires an additional step to extract the cost for the LSP because a part of their routes is paid by the taxi company and the other part has to be paid by the LSP. The tables in Appendix 6.1, 6.2, 6.3 and 6.4
contain the information we need for the assessment of the benefit for the LSP. The total benefit for the LSP is shown in column 18 of these tables. This benefit is the cost reduction between the model without scheduled lines and the model with scheduled lines. In the former model the cost was comprised of only the truck cost (column 5). In the latter model the cost is comprised of the truck cost (column 9), the cost for using scheduled lines (column 16) and the cost for using taxis (column 12).

**Tuesday experiment.** For the Tuesday dataset there is no benefit when the scheduled line (SL) cost is €3 per trolley in the case of 15 taxi passengers and there is no benefit when the SL cost is €2 in the case of 45 taxi passengers. The cost of a SL is too high and therefore no scheduled lines are beneficial. When the cost of a scheduled lines per trolley is €2 or lower than most of the solutions improved when using scheduled lines. As can be expected the benefit for the LSP is greatest when the truck cost is high and the taxi cost is low. When we look at the parameter setting of €0.8 per truck km then we see an average cost reduction of 34% when the SL cost is €0 and a 17% average cost reduction for a SL cost of €0.5, a 10% cost reduction when the SL cost is €1 and no cost reduction when the SL cost is €2. When a truck costs €0.9 per km this is respectively 26%, 18%, 14% and 6%. This show a great reduction in cost.

In the routing solution it can be seen that for the requests with their delivery location in Helmond most times use a scheduled lines except when the price of the SL per trolley reaches €3. This shows that the greatest benefit can be reached when there are clusters of delivery location in different municipalities. Besides the Helmond cargo requests some of the cargo requests with their delivery location in Eindhoven also were serviced by scheduled lines. The number of used scheduled lines for Eindhoven cargo request lies between 0 and 6 when the truck cost is €0.9 and the number lies between 0 and 5 when the truck cost is €0.8. This indicates that there is also potential for clusters which lie relatively close to the depot.

**Wednesday experiment.** This dataset has more cargo request deliveries in both Eindhoven and Helmond. In the Tuesday dataset there was no benefit for the LSP when the SL cost reached €3 per trolley. However this dataset does not result in benefit for the LSP when the SL cost reaches €2 for both the cases with 15 and 45 taxi passengers. The reason is that this set requires more scheduled lines in order to be beneficial. In the Tuesday dataset the lowest number of chosen scheduled lines while still being beneficial compared to the old system was 8 scheduled lines but in this dataset the minimum used schedules lines is 28. Because more scheduled lines are chosen the impact on the cost increase for these scheduled lines is also higher. The reason for the higher required scheduled lines is that a bigger portion of all cargo requests lie in a cluster like Helmond.

As in the Tuesday dataset the most benefit is reached when the truck cost is €0.9 and the taxis cost is €0.3. This can be expected because every additional reduction in truck kilometers is relatively large because of this parameter setting and every additional increase in taxi cost is relatively low. The cost reduction for this dataset is larger when the SL cost is low and lower when the SL cost is high than in
the Tuesday dataset. When a truck costs €0.8 per km then the average cost reduction is 34%, 15% and 1% for respectively the SL cost of €0, €0.5 and €1. For the truck cost of €0.9 per km this is respectively 37%, 19% and 3%. Table 4 shows the overview of the comparison between all four datasets. In this table you can see that when the SL cost is around €1 per trolley the Tuesday dataset receives more benefit then the Wednesday dataset and the Wednesday dataset receives more benefit than the Tuesday dataset when the SL cost is lower.

For almost all of the parameter setting there has been found a significant benefit for the LSP indicating potential benefit when companies integrate scheduled lines for cargo deliveries.

For both the Tuesday and the Wednesday dataset it has been shown that the average cost reduction is the highest for the case of 45 taxi passengers. This is not a surprise because the heuristic algorithm was able to choose between more routes.

6.2.4 Taxi and bus company benefit from Cargo Hitching
For the taxi company I have looked at what the increase of distance is when they implement the Cargo Hitching. This increase in distance has been compared to the additional income from transporting the cargo. The last thing is that I have looked at how much delay the passengers have. For the bus company I have looked at the additional income because in this case study they do not have additional cost. The overview of the total benefit for all parameter settings can be found in Appendix 6.1, 6.2, 6.3 and 6.4.

6.2.4.1 Taxi company benefit
The information required for assessing the taxi benefit can be found in the tables of Appendix 6.1, 6.2, 6.3 and 6.4. In column 19 the total benefit can be found for the taxi company. This total benefit is the result of the difference between the traveled distance by taxis in the system without scheduled lines and the new system with scheduled lines. For the difference in distance traveled the taxi company has a cost of €0.125 per kilometer. The LSP however had to pay €0.3 or €0.4 per additional kilometer. The difference between the additional income and the additional cost for the taxi company is the total benefit. The taxi distance traveled in the system without scheduled lines can be found in column 6 and the cost for the taxi company for this distance can be found in column 8. The new taxi distance traveled in the system with scheduled lines can be found in column 11. The price that the LSP has to pay to the taxi company can be found in column 12 and the total cost for the taxi company can be found in column 13.

Tuesday experiment. The result of this experiment is that as long as the LSP chooses to use a scheduled line that the taxi company will benefit. This is logical because all additional costs are covered plus an additional margin per kilometer. However the price they ask for every additional kilometer has an effect on how many scheduled lines will be used. The average additional income when they ask €0.3 per km is €5.10 over the two taxi passenger sets and the parameter setting for SL costs of €0, €0.5, €1 and €2. This additional income is €5.80 when they ask €0.4 for every additional km. This would indicate that asking a price of €0.4 would result in more income than asking €0.3. But this is not always the case.
When the SL cost is lower than €1 then asking €0.4 would provide more benefit but when the SL cost is higher then this will results in less income. For the taxi company it is therefore important to look at what the SL cost is.

**Wednesday experiment.** The Wednesday dataset shows that there is a big difference for the additional taxi income between only using 15 taxi passengers and using 45 taxi passengers. The reason for this is that more scheduled lines have to be used in order to outsource the delivery towards Helmond. Because of this higher number of scheduled lines also more taxis must be used. The set of 45 taxi passengers therefore shows a lot more possibility for integrating cargo and results in less dramatic increase in taxi distance. The impact of the taxi and the SL cost however is a lot lower. The Wednesday dataset shows a different scenario than the Tuesday dataset in this case. It does not significantly matter how much the taxi company asks per additional kilometer traveled because the LSP will use around 28 to 30 line in the case of 15 taxi passengers and 38 replicated scheduled lines in the case of 45 taxi passengers. This holds until the prices of a replicated scheduled line reaches €2 per trolley. This shows that it is most profitable for the taxi company to ask a price of €0.4 per additional kilometer traveled.

In the Tuesday datasets you can conclude that the asking price for every additional kilometer traveled must be based on the price that the bus line ask for servicing a trolley. But the Wednesday dataset show that they can ask the highest price and this does not influence the use of replicated scheduled lines by the LSP. Both datasets also show that there is potential for a taxi company to integrate cargo deliveries as part of their service. Another important point to mention is that only 1 out of 10 passenger requests their slack was used. Which means that only 1 out of 10 passenger request are dropped off at their delivery location with a maximum delay of ten minutes.

**6.2.4.2 Bus line company benefit**

For the benefit of the busses it is important to maximize the total additional income because there is only minor additional cost like handling cost. I am not going to discuss at the parameter setting when a bus line has a price of €0 per trolley. Obviously this parameter setting resulted in the highest number of used bus lines for the transport of cargo but I cannot say how much income they could generate if they would receive subsidy or related income.

The total additional income for the bus line company can be found in column 16 of the tables in Appendix 6.1, 6.2, 6.3 and 6.4. For the Tuesday dataset the optimal price is €1 per trolley. However when the bus line asks €1 per trolley in the Wednesday dataset then in half of the cases no additional income would be generated. In Table 4 and overview of the total additional income can be found of the bus line (column 12, 13, 14 and 15). In this table can be found that the average additional income over both datasets is greatest when they ask a cost of €1 per trolley with an average of €10.5 instead of €8.6 when they ask €0.5.
6.2.5 Society benefit from Cargo Hitching

In order to assess the benefit for the society I have looked at the total distance reduction of the trucks. This will be put in contrast with the increase in taxi distance but with a different weight for each of the vehicle types their kilometers. The city of Eindhoven receives congestion issues easier as a result of trucks and less from taxis, therefore the weight of a kilometer with a truck will be higher than the weight of a taxi.

Table 4 shows the overview of total reduction in kilometers. Column 16, 17, 18 and 19 present this total reduction with a weight on the trucks kilometers. In general trucks are more stressful for the environment and can cause congestion easier than taxis. Therefore I used weight of 2 for the trucks, this means that the kilometers traveled by the truck have a bigger influence on the total reduction by the system with scheduled lines.

The biggest reduction in total kilometers is reached when the cost for the LSP of taxi is €0.4 per kilometer. The reasoning is that the LSP will use the scheduled lines only for delivery locations which significant reduce the total kilometers if they are serviced by a scheduled line. The average kilometer reduction of all four datasets is 30.1 km for taxi cost of €0.3 and 29.7 km for a taxi cost of €0.4. The SL cost is again also a big influence on the total reduction. When the SL cost of 1 trolley equals €0.5 then the largest reduction in kilometers is achieved. The average kilometer reduction of all four datasets together is 39.1 kilometers. The average kilometer reduction of both datasets together is therefore highest when the SL cost is €0.5 and the taxi cost is €0.4. This reduction equals 40.2 kilometers instead of 38 kilometers when the taxi cost is €0.3. These results also shows great potential for the society in terms of congestion and pollution decrease.

Table 4 Overview of benefit for multiple stakeholders

<table>
<thead>
<tr>
<th>Day of dataset # taxi passengers</th>
<th>LSP cost reduction</th>
<th>Taxi benefit</th>
<th>SL benefit</th>
<th>Society benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COST</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(€)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(€)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(€)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(€)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(€)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(km)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(km)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>(km)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In order to assess the benefit for the society I have looked at the total distance reduction of the trucks. This will be put in contrast with the increase in taxi distance but with a different weight for each of the vehicle types their kilometers. The city of Eindhoven receives congestion issues easier as a result of trucks and less from taxis, therefore the weight of a kilometer with a truck will be higher than the weight of a taxi.

Table 4 shows the overview of total reduction in kilometers. Column 16, 17, 18 and 19 present this total reduction with a weight on the trucks kilometers. In general trucks are more stressful for the environment and can cause congestion easier than taxis. Therefore I used weight of 2 for the trucks, this means that the kilometers traveled by the truck have a bigger influence on the total reduction by the system with scheduled lines.

The biggest reduction in total kilometers is reached when the cost for the LSP of taxi is €0.4 per kilometer. The reasoning is that the LSP will use the scheduled lines only for delivery locations which significant reduce the total kilometers if they are serviced by a scheduled line. The average kilometer reduction of all four datasets is 30.1 km for taxi cost of €0.3 and 29.7 km for a taxi cost of €0.4. The SL cost is again also a big influence on the total reduction. When the SL cost of 1 trolley equals €0.5 then the largest reduction in kilometers is achieved. The average kilometer reduction of all four datasets together is 39.1 kilometers. The average kilometer reduction of both datasets together is therefore highest when the SL cost is €0.5 and the taxi cost is €0.4. This reduction equals 40.2 kilometers instead of 38 kilometers when the taxi cost is €0.3. These results also shows great potential for the society in terms of congestion and pollution decrease.
6.2.6 System benefit – cost price parameter setting

The system benefit is the total benefit if all the companies perform their service at their cost price. The easiest way to explain this is to imagine an intermediate party which controls the system. They receive the reduction in cost of the LSP trucks and they pay the cost of the additional taxi distance. The benefit that they have left afterwards is the total amount of money which has to be split up between the three companies (LSP, taxi and bus line company). For this situation the heuristic algorithm has been used with €0.8 and €0.9 variable cost per km for a LSP truck, €0.125 variable cost per km for a taxi and €0 for the use of a scheduled line. Table 5 presents an overview for both the Tuesday and Wednesday dataset with 45 taxi passengers. In this table you can see that for example the Wednesday set with 45 taxi passengers at LSP truck cost price of €0.8, taxi cost price of €0.125 and no cost for a scheduled line that the system benefit equals €34.40. This amount can be divided among the three different companies. In that specific solution in total 28.73 trolleys travel with a scheduled line and if for example the bus line company would ask €1 per trolley then there would be €5.67 left for the taxi company and the LSP. The system benefit shows the benefit that can be achieved with Cargo Hitching if the companies do not try to achieve their own maximal benefit.

Table 5 System benefit. Taxi passengers size = 45

<table>
<thead>
<tr>
<th>Sl cost (€)</th>
<th>Truck cost (€)</th>
<th>Taxi cost (€)</th>
<th>Old situation</th>
<th>LSP (€)</th>
<th>Taxi (€)</th>
<th>System Benefit (€)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuesday</td>
<td></td>
<td></td>
<td>LSP cost</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>0.80</td>
<td>0.13</td>
<td>61.60</td>
<td>83.94</td>
<td>1.44</td>
<td>49.26</td>
</tr>
<tr>
<td>0.00</td>
<td>0.90</td>
<td>0.13</td>
<td>69.30</td>
<td>83.94</td>
<td>1.62</td>
<td>56.78</td>
</tr>
<tr>
<td>Wednesday</td>
<td></td>
<td></td>
<td>LSP cost</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.00</td>
<td>0.80</td>
<td>0.13</td>
<td>61.60</td>
<td>83.94</td>
<td>22.24</td>
<td>34.44</td>
</tr>
<tr>
<td>0.00</td>
<td>0.90</td>
<td>0.13</td>
<td>69.30</td>
<td>83.94</td>
<td>25.02</td>
<td>39.36</td>
</tr>
</tbody>
</table>

6.3 Evaluation of heuristic

For the evaluation of the heuristic I have used the mixed integer linear problem formulations (MILP) from chapter 4.2.1 and 4.2.2. The formulations are used to describe the two systems. The first system is the case without the possibility of using scheduled lines and the second system is the case where the use of scheduled lines if permitted. An optimal solution can be created with the help of exact methods. The first MILP from chapter 4.2.1 is less complex and closely resembles the current LSP situation without transshipments and scheduled lines. The formulation describes one type of vehicle, one type of request and no scheduled lines. The type of vehicle is a LSP truck with capacity of 40. I chose capacity of 40 because over 90% of the cargo requests required a maximum size truck of 40 trolleys capacity for the delivery. The type of requests is the cargo requests of the LSP. This exact method is used to create a lower limit for the heuristic. The lower limit is used to indicate the increased performance of the heuristic. This method could solve up to 20 cargo request in a relatively fast manner of two days. More request were possible but the model tended to exponentially increase in CPU time and required a lot of system memory.

The MILP from chapter 4.2.2 is far more complex and is the desired system which includes transshipments and scheduled lines. This formulations involves two types of requests, two types of
vehicles and scheduled lines which are replicated for every cargo request. This dramatically increases the complexity of the model. The largest possible size which my laptop could solve was 8 cargo requests, one passenger request, one scheduled line, one truck, one taxi and two depots. I tried to change the number of cargo requests from 8 to 9 but the laptop ran out of memory and could not solve the problem. The solution with 8 cargo requests is not sufficient to fully evaluate the performance of the heuristic, but I was not able to compute a solution with more requests.

The heuristic has been evaluated with a lower bound created by the mixed integer linear program formulation for the system without scheduled lines. The instances used for this experiment are the same as used for the heuristic. For the heuristic I have used two instances, the first one contains 12 cargo requests and the second one contains 20 cargo requests. These experiment have been executed with different cost parameters as described in the experimental design. The results of the lower bound versus the heuristic are shown in Table 6.

If the heuristic would perform badly and is unable to insert scheduled lines the cost reduction would be 0% in all cases. If the heuristic would insert scheduled lines which are not beneficial then the cost reduction could become negative. This is not the case in any of the parameter setting. The average reduction over all the parameter setting for the Tuesday dataset is 15% in cost and for the Wednesday dataset this is 18% which is a good result.

Another very important aspect of the heuristic is the CPU time. For a LSP it can be important to create a new routing solution within a short amount of time. The exact method took 166 min in order to create a solution for six cargo requests and 1 passenger request. A LSP has a lot more requests than 6 and therefore an exact method is not a possibility. The heuristic algorithm can be the solution for this problem. As can be found in the table of Appendix 6.1, 6.2, 6.3 and 6.4, the average duration of the heuristic algorithm is 18 second for 12 request with 15 taxi passengers and 1:42 minutes for 45 taxi passengers. For the Wednesday dataset with 20 cargo request and 15 taxi passengers the heuristic took an average of 2:40 minutes and for 45 taxi passengers this was 5:07 minutes. For the algorithm it is also
important what the demand sizes are of these requests because it splits cargo requests which have a
demand size larger than the taxi capacity. The Tuesday dataset had in total 15 cargo requests after
splitting and the Wednesday dataset had in total 29 cargo requests after splitting. The duration of the
heuristic is significantly lower than the exact method and is able to create a routing solution for a large
number of request within minutes.

Chapter 7
Conclusions and Future Work

In this conclusion I will first start with answering the sub research question stated in section 3.3. After
these questions I will present the general conclusions of the case study. In section 7.3 I will elaborate
further on limitations on the case study and in section 7.4 I will present my view on future research.

7.1 Sub research questions

*Is it possible to capture the full case study situation in one linear mathematical model?*

To a great extent it is indeed possible to capture the case study in a linear mathematical model. However,
there are parts of the case study which have not been captured in the model. One of the examples is that
a LSP truck sometimes leaves a trailer behind at a customer and then drives further into the city center.
On its way back the LSP truck will pick up the trailer again. It does this because some streets are not
accessible by a truck with trailer. Another example is that the LSP sometimes uses soft time windows
in consultation with the customer. Small examples like these are not accounted for in the mathematical
model but are only a small portion of the delivery system. The reason that they are not accounted for is
that they increase the complexity of the model without having a big impact.

*Is it possible to find the optimal solution to this model using an exact algorithm?*

Yes this is indeed possible, but not in reasonable amount of time. If a company finds a way to decrease
the run time or decides to plan far ahead into the future then there is still a problem with the required
computer memory. If my laptop with an 8 GB RAM is not even sufficient for 6 cargo requests and more
than 1 passenger requests then it will be incredibly hard to use this method for large instances.

*Is the exact algorithm fast enough for companies like Lekkerland?*

I believe that using a pure exact method is not applicable for companies who will have to make a
planning every day. This requires that a method finds a solution within a few minutes to hours and for
a large dataset this will simply not be possible without an extreme super computer.

*Which heuristics method can be used to create a good and fast solution to the mathematical model but
not necessarily optimal?*

A heuristic method which creates a good solution for this problem will need to search large
neighborhoods and then checks if the solution has improved. Examples of these methods are Large
Neighborhood Search method and Simulated Annealing method. The reason is that the PDP-SL most of
the time receives the biggest benefit when multiple scheduled lines are used, I elaborated on this earlier in chapter 5. In short the reason for this is that one scheduled line is a large part of the time not enough to create a decrease in cost. Ghilas et al. (2014) for example used an Adaptive Large Neighborhood Search and found very promising results for the PDP-SL.

**How good do these heuristics perform with respect to the optimal solution?**

Ghilas et al. found that the heuristic came very close to the optimal solution within a few percentages. For this case study it is very hard to answer as I was not able to find upper bounds for large datasets. Therefore I cannot say much about the performance of larger problems. However I did established a lower bound with an exact method based on the old model. The heuristic did perform better than this Lower bound. I will further elaborate on this question in chapter 6.3.

**Which transshipment hub locations are suitable for the case study?**

The location of the transshipment hubs are very dependent on which dataset I use. One important thing to mention is that it is beneficial to have a transshipment hub close to the LSP depot. In all of the datasets the scheduled lines from these transshipment hubs were most frequently in the best solution. Further it is important that the other transshipment hubs lie in the center of the cities of the cargo request delivery locations. From these centers most of the delivery locations can be reached by taxis. I chose for example a transshipment hub in Eindhoven and Helmond because these two hubs lie in the city center of delivery location clusters.

**What will be the impact when some parameters change? Think for example of the cost for using a scheduled line or the capacity of a taxi.**

The impact varies and in order to answer this question I have analyzed a number of datasets which are discussed in chapter 6. The results have shown that taxi companies and bus companies can optimize their profit by setting the prices to a certain height. If we would look at only the Tuesday and Wednesday dataset then we see that the optimal asking price for the bus company is €1 per trolley. If the bus company would charge €2 than the income is on average less than half of what would be the additional income when they charge €1 per trolley. If they would charge €0.5 per trolley then the additional income would be a little above the €2 per trolley but still half of the additional income of €1 per trolley. For the taxis it is most beneficial to ask €0.4 per additional kilometer of a taxi. This is true for all cases if we would look at the average of both datasets. However only looking at the Tuesday dataset is show that for a higher scheduled line cost it would be more beneficial to charge €0.3 than €0.4 per additional kilometer.

### 7.2 General Conclusion

In this thesis I introduced a case study which has been used to test the potential of the Pickup and Delivery problem with the option to include scheduled lines and transshipment (PDP-SL) in the real world. The case study tests this new PDP-SL system and compares it to the current delivery system. In
order to experiment with this case study I have translated the current LSP delivery system and the PDP-SL system into two mathematical formulations. Two exact methods were used to create routing solutions according to the mathematical formulation. These two solution are used as a lower and upper bound for the PDP-SL system.

Further a heuristic method was developed which could solve larger problem within a reasonable amount of computational time. The heuristic was specifically developed for this case study and uses greedy and tabu search element for the construction of an initial solution. Further it uses elements from the Tabu Search and Large Neighborhood Search in order to search the neighborhood spaces. At the end of the heuristic also 2-opts and vehicle switching were used to further improve the solution.

The heuristic was tested with data of a real world logistic service provider (LSP). The data extracted from this LSP included information about cargo delivery requests they have received and information about their vehicle fleet. This information also included time windows, service times and location of pickup and delivery. The data about passenger requests was randomly generated based on a normal distribution. The data about the taxis, taxi depot locations and scheduled lines locations was created using a heuristic which finds the optimal locations from the real world. These optimal locations were based on real locations like bus stop, taxi depots and city center stations. The results from the heuristic show that there is a potential in the PDP-SL for every stakeholder in the system. This includes the LSP itself, the taxi company, the scheduled line company and the society. For the LSP, taxi and scheduled line company the potential benefit was financial and the society had a potential benefit in a decrease in congestion and pollution.

The performance of the heuristic was then evaluated with the lower bound created by the exact method for the current system. The heuristic did perform relatively well and in most parameter setting improved the optimal solution from the exact method of the old system. The exact method of the upper bound could not solve problems of meaningful size. This was due to the complexity of the PDP-SL. In these small problems the heuristic performed well but as the problem size will increase the gap between the upper bound and the heuristic will also increase. This means that there is still improvement left for the heuristic.

Even though the heuristic can be improved the results already show significant benefit from integrating scheduled lines and transshipment hubs. A large part of the route of cargo requests can be performed by scheduled lines without increasing the additional taxi distance too much. It is also shown that the benefit for the taxi company and the scheduled lines company can be manipulated by changing the price of their service. There is an optimal price for their service which maximize their benefit and it is interesting for these companies to do more research on this topic if the integration of scheduled lines would be implemented. My suggestion is to perform more case studies and real life test. With this thesis I have shown that there is indeed a great possibility for different type of transportation companies to cooperate.
but it is hard to generalize. The case study involved only one company with its own specific type of delivery and specific type of customers. Even within this company I was able to only look at one distribution center and not even all of its clients. I also suggest that the different type of companies would do a joint research on what the optimal charges would be in order to get the most out of the system for all of the stakeholders as I have seen in the results that these charged have a good impact on the optimal solution.

In this thesis I have examined multiple datasets which include real world data, but much more is required in terms of future research. I will present a number of these future research areas in the next section.

7.3 Limitations

A first limitation is that I made assumptions to speed up the exact solution method. My initial model included that passenger request are allowed to have a delay of 10 minutes as a result of intermediate pickups or delivery of a cargo request. As a consequence with every minute of delay a penalty incurred in the objective function. Unfortunately this increased the duration significantly and the penalty was not often used, therefore I chose to eliminate this penalty from the model.

Another assumption was that a taxi could hold up to 1 trolley of cargo request at a time. This may seem rather large and it assumes that the passenger request does not use any of that capacity. The reason I did not lower this capacity is because the model would increase in complexity once again and therefore the system runs out of memory earlier or the CPU time would increase.

For the scheduled lines I have also made assumptions about the available capacity. The current scheduled line vehicles from my case study do not have an available compartment for trolleys in the real world. I assumed that a maximum of ten trolleys were able to fit in the scheduled line vehicles outside of the rush hours.

7.4 Future work

There are multiple aspects in my research which can be improved by people with more ready knowledge about programming or route optimization or by doing more extensive research. These people could create a more efficient program which would be able to handle more complex and bigger problems. But there are also aspects which fell out of the scope of my thesis which are important to research before implementing such system.

The first problem is that the heuristic its upper bound could only be tested with a very limited number of requests, vehicles and scheduled lines. I believe that people with more experience in creating mixed integer linear problem formulations for complex systems can make a more efficient formulation which will be able to test greater data sets or people which have access to computer technology which is more capable than my laptop.
The second future research aspect regards my assumptions of the real world about subjects which are not yet researched or implemented. For example, the transshipments between trucks, taxis and transfer hubs now include a service time which is just an approximation. No implemented system exists which is able to transfer between scheduled lines vehicles and trucks/taxis in an efficient way. At this moment there are concepts of these transfer hubs but no implementation yet. Also the safety is a research topic yet to be researched. How to make sure that no cargo will get stolen while traveling on a public scheduled line or taxi.

A third important research topic is to determine the prices for using a scheduled line or taxi. These are not yet determined in the real world and therefore I looked at multiple different values for these prices. Maybe some governments are willing to give subsidy for such projects which decrease carbon footprint or congestion in cities.

A fourth future research topic is that in practice it is common to have certain delays in for example departure times of scheduled line vehicles, taxi passengers which are late or congestion/accidents. My heuristic algorithm does not include these possibilities and could therefore create problems when implemented. How to deal with these deviations needs to be researched and could also be included in a heuristic algorithm.

A fifth research topic is the research on how do taxi passengers perceive the possible delay they receive as a result of cargo delivery. Do they require certain financial incentives and if yes how much? Another research topic is how to deal with a high unexpected number of passengers in a bus. When there are too many passenger which was unexpected than some if not all cargo request will have to travel differently.

One of the most important future research topics are tests in practice. These tests in practice should involve cases which for example try to efficiently transfer cargo between hubs and busses, or how to customize a bus in order for cargo trolleys to fit, or what is the easiest way to load and unload a taxi with cargo. These are all points which can be studied in literature but ultimately have to be tested in the real world.

Out of the box ideas like the Cargo Hitching between public and private companies is a very nice example of how we can improve the current way of transport. Many other out of the box examples are already being researched and research need to keep on doing this. If we really want to change the way how transportation is operating then we need more of these ideas and more research on them. I believe that this is one of the most important future research focus.
Chapter 8

Bibliography


### Appendix 4.1

*Table 7 Tightening of vehicle routing decision variable $x_{ij}^v$*

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>v</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>P1</td>
<td>V1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>V2</td>
</tr>
<tr>
<td>P2</td>
<td>V1</td>
<td>V2</td>
</tr>
<tr>
<td>D1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>v</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>P1</td>
<td>V1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>V2</td>
</tr>
<tr>
<td>P2</td>
<td>V1</td>
<td>V2</td>
</tr>
<tr>
<td>D1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>v</th>
</tr>
</thead>
<tbody>
<tr>
<td>D2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>v</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>P1</td>
<td>V1</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>v</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>P1</td>
<td>V1</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>V1</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>V2</td>
<td></td>
</tr>
</tbody>
</table>
### Appendix 4.2

**Table 8 Tightening of request routing decision variable $y_{ij}^r$**

<table>
<thead>
<tr>
<th>i</th>
<th>j</th>
<th>r</th>
<th></th>
<th>i</th>
<th>j</th>
<th>r</th>
<th></th>
<th>i</th>
<th>j</th>
<th>r</th>
<th></th>
<th>i</th>
<th>j</th>
<th>r</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>P1</td>
<td>r1</td>
<td>✓</td>
<td>P2</td>
<td>P1</td>
<td>r1</td>
<td></td>
<td>P1</td>
<td>P1</td>
<td>r1</td>
<td>✓</td>
<td>P2</td>
<td>P1</td>
<td>r1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>r1</td>
<td></td>
<td></td>
<td>D2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D2</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>r1</td>
<td></td>
<td></td>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>P1</td>
<td>r1</td>
<td>✓</td>
<td>T1</td>
<td>P1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>P1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T2</td>
<td>P1</td>
<td>r1</td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>r1</td>
<td></td>
<td></td>
<td>D2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D2</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>P1</td>
<td>r1</td>
<td>✓</td>
<td>T1</td>
<td>P1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>P1</td>
<td>r1</td>
<td>✓</td>
<td>T2</td>
<td>P1</td>
<td>r1</td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D1</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D2</td>
<td>r1</td>
<td></td>
<td></td>
<td>D2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>D2</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T1</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
<td>T2</td>
<td>r1</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
<td>W</td>
<td>r1</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
<td></td>
<td>r2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix 4.3

\[
\begin{align*}
\text{min} & \quad \theta \sum_{i,j \in \mathcal{K}} \sum_{v \in \mathcal{V}} c_{ij} y_{ij} \\
& \quad + \eta \sum_{i \in \mathcal{T}} \sum_{j \in \mathcal{K}, v \in \mathcal{V}} \sum_{w \in \mathcal{Z}} q_{ij}^{vw} \\
\text{subject to} & \quad \sum_{j \in \mathcal{K}} \sum_{v \in \mathcal{V}} y_{ij} = 1 & \forall j \in \mathcal{R}^i \\
& \quad \sum_{v \in \mathcal{V}} y_{ij} = 1 & \forall v \in \mathcal{V} \\
& \quad \sum_{i \in \mathcal{T}} \sum_{j \in \mathcal{K}, v \in \mathcal{V}} q_{ij}^{vw} = 1 & \forall r \in \bar{\mathcal{T}} \\
& \quad \sum_{j \in \mathcal{K}} y_{ij} - \sum_{j \in \mathcal{K}} x_{ij} = 0 & \forall i \in \mathcal{N}, v \in \mathcal{V} \\
& \quad \sum_{j \in \mathcal{K}} x_{ij} = x_{i}^e & \forall r \in \bar{\mathcal{T}}, i \in \mathcal{K}^3 \\
& \quad \sum_{j \in \mathcal{K}} \sum_{v \in \mathcal{V}} x_{ij} = \sum_{j \in \mathcal{K}} \sum_{v \in \mathcal{V}} y_{ij} & \forall i \in \mathcal{T} \\
& \quad \sum_{j \in \mathcal{K}} \sum_{v \in \mathcal{V}} x_{ij} \geq \sum_{j \in \mathcal{K}} \sum_{v \in \mathcal{V}} x_{ij} - \sum_{j \in \mathcal{K}} \sum_{v \in \mathcal{V}} x_{ij} & \forall i \in \mathcal{T} \\
& \quad y_{ij} \geq 1 - c_{ij} + \delta_j - M_{ij}^1 (1 - y_{ij}) & \forall r \in \bar{\mathcal{T}}, i, j \in \mathcal{K}^3 \\
& \quad \beta_i \geq 1 - c_{ij} + \delta_j - M_{ij}^1 (1 - \sum_{j \in \mathcal{K}} y_{ij}) & \forall i \in \mathcal{N}, j \in \mathcal{R}^3 \\
& \quad \alpha_i \leq 1 - \alpha_j - \delta_i \leq 1 - u_i & \forall i \in \mathcal{R}^1 \\
& \quad \sum_{w \in \mathcal{Z}} q_{ij}^{vw} = y_{ij} & \forall r \in \bar{\mathcal{T}}, (i, j, w) \in \mathcal{F}^r \\
& \quad y_{ij} \leq M_{ij}^2 (2 - q_{ij}^{vw} - y_{ij}) & \forall r \in \bar{\mathcal{T}}, (i, j, w) \in \mathcal{F}^r, w \in \mathcal{Z}^j \\
& \quad y_{ij} \geq - M_{ij}^2 (2 - q_{ij}^{vw} - y_{ij}) & \forall r \in \bar{\mathcal{T}}, (i, j, w) \in \mathcal{F}^r, w \in \mathcal{Z}^j \\
& \quad \sum_{j \in \mathcal{K}} \sum_{v \in \mathcal{V}} y_{ij} \leq k_{ij} & \forall i, j \in \mathcal{E}, w \in \mathcal{K}^j \\
& \quad y_{ij} \leq M_{ij}^1 (1 - \sum_{j \in \mathcal{K}} y_{ij}) & \forall r \in \bar{\mathcal{T}}, i \in \mathcal{T} \\
& \quad y_{ij} \geq - M_{ij}^1 (1 - \sum_{j \in \mathcal{K}} y_{ij}) & \forall r \in \bar{\mathcal{T}}, i \in \mathcal{T} \\
& \quad y_{ij} \leq M_{ij}^1 (1 - \sum_{j \in \mathcal{K}} y_{ij}) & \forall r \in \bar{\mathcal{T}}, i \in \mathcal{R}^1 \\
& \quad y_{ij} \geq - M_{ij}^1 (1 - \sum_{j \in \mathcal{K}} y_{ij}) & \forall r \in \bar{\mathcal{T}}, i \in \mathcal{R}^1 \\
& \quad y_{ij} \leq M_{ij}^1 (1 - y_{ij}) & \forall r \in \bar{\mathcal{T}}, i \in \mathcal{T}, j \in \mathcal{O}^j \\
& \quad y_{ij} \geq - M_{ij}^1 (1 - y_{ij}) & \forall r \in \bar{\mathcal{T}}, i \in \mathcal{T}, j \in \mathcal{O}^j \\
& \quad x_{ij} \in [0, 1] & \forall i \in \mathcal{N}, v \in \mathcal{V} \\
& \quad y_{ij} \in [0, 1] & \forall i \in \mathcal{K}, r \in \bar{\mathcal{T}} \\
& \quad \alpha_i \in \mathcal{R}^+ & \forall i \in \mathcal{N} \\
& \quad \beta_i \in \mathcal{R}^+ & \forall r \in \bar{\mathcal{T}}, (i, j) \in \mathcal{F}^r, w \in \mathcal{Z}^j \\
& \quad q_{ij}^{vw} \in [0, 1] & \forall r \in \bar{\mathcal{T}}, (i, j) \in \mathcal{F}^r, w \in \mathcal{Z}^j
\end{align*}
\]

Figure 10 Mathematical formulation of Ghilas et al. 2015
Appendix 4.4

The objective function is the total operating costs, including traveling cost of the PD vehicles (1) and the total cost of using the scheduled lined services (2). Constraints (3) assure that all nodes related to the requests (pickup and drop-off nodes) are visited exactly once. Constraints (4) make sure that each vehicle leaves its depot at most once, and (5) assure that each replicated transfer node is visited at most once. Flow conservation for PD vehicles is considered in constraints (6). Constraints (7) assure flow conservation for the paths of each request. Constraints (8) ensure that if a request uses a scheduled line, a PD vehicle should pick it up/drop it off at a station-hub related to that specific scheduled line. Constraints (9) force the capacity of each PD vehicle is not exceeded. Scheduling for each request and each PD vehicle is considered in constraints (10) – (12). Constraints (13) assure that the pickup node is visited before the drop-off node of each request and (14), (15) force the time windows to be respected. Constraints (16) – (18) assure that if a request uses a scheduled line, it departs at a scheduled departure time. Constraints (19) ensure that the package carrying capacity of the scheduled lines is not exceeded. Constraints (20) – (27) assure the synchronization between requests and PD vehicles. The domains of the decision variables are defined in (28) – (33). Note that $M_t^{ij}$ can be substituted by $(u_i + s_i + s_j + c_{ij})$ and $(u_i + s_i)$ can be used instead of $M_t^{ij}$.

Figure 11 Constraints of the Ghilas et al. 2015 mathematical formulation
Appendix 5.1

Algorithm 1: Phase 1 of the metaheuristic – initial solution for LSP

**input:** Set of Lekkerland vehicles \(setV1\) and LSP requests \(setP1\)

**output:** A feasible vehicle routing solution for LSP trucks in the variable \(newX\)

```plaintext
while (run) do
    Initialize requests left list \(L\) // list of passengers requests
    Initialize empty tabu requests list \(T_r\) // list of tabu requests
    Initialize empty tabu vehicle list \(T_v\) // list of tabu vehicles
    \(newX \longleftarrow NULL\)
    \(bestX \longleftarrow NULL\)
    \(i \longleftarrow depot\)
    \(newVehicle \longleftarrow true\)
    if (newVehicle) then
        \(v \longleftarrow selectVehicle(newX)\) // select vehicle with largest capacity
        \(d \longleftarrow closestDeliveryNode(i, L, T_r)\) // select closest request
        \(p \longleftarrow pickup location of d\)
        \(newX \longleftarrow insertRoute(newX, v, i, p, d)\) // insert request in route
        if (CheckCapacity(v, newX)) then
            if (CheckTime(v, newX)) then
                \(bestX \longleftarrow newX\) /* if constraints are met then save current solution */
                \(L \longleftarrow L \cap d\) in bestX and remove request from \(L\)
            else
                \(T_r \longleftarrow T_r \cup d\) /* if constraints not met, add request to tabu list \(T_r\) */
                \(newX \longleftarrow bestX\)
            else
                \(T_r \longleftarrow T_r \cup d\) /* if constraints not met, add request to tabu list \(T_r\) */
                \(newX \longleftarrow bestX\)
                \(newVehicle \longleftarrow true\) /* if no more options left for vehicle but there are requests left, select new vehicle and add current vehicle to tabu vehicle list \(T_v\) */
        else
            \(newX \longleftarrow bestX\) and reset \(newX\) to last known feasible solution */
    else
        \(T_r \longleftarrow T_r \cup d\) /* if no more requests left, then start next phase */
        \(newX \longleftarrow bestX\)
    if (L.size == 0) then
        \(run \longleftarrow false\)
    else
        \(T_v \longleftarrow T_v \cup v\)
        \(T_r \longleftarrow NULL\)
        \(newVehicle \longleftarrow true\)
    if (setV.size == T_v.size) then
        \(run \longleftarrow false\) /* if no more vehicles are left then stop algorithm */
```

1. **Appendix 5.1**
2. **Algorithm 1:** Phase 1 of the metaheuristic – initial solution for LSP
3. **Input:** Set of Lekkerland vehicles \(setV1\) and LSP requests \(setP1\)
4. **Output:** A feasible vehicle routing solution for LSP trucks in the variable \(newX\)
5. while (run) do
6.   Initialize requests left list \(L\) // list of passengers requests
7.   Initialize empty tabu requests list \(T_r\) // list of tabu requests
8.   Initialize empty tabu vehicle list \(T_v\) // list of tabu vehicles
9.   \(newX \leftarrow NULL\)
10. \(bestX \leftarrow NULL\)
11. \(i \leftarrow depot\)
12. \(newVehicle \leftarrow true\)
13. if (newVehicle) then
14.   \(v \leftarrow selectVehicle(newX)\) // select vehicle with largest capacity
15.   \(d \leftarrow closestDeliveryNode(i, L, T_r)\) // select closest request
16.   \(p \leftarrow pickup location of d\)
17.   \(newX \leftarrow insertRoute(newX, v, i, p, d)\) // insert request in route
18.   if (CheckCapacity(v, newX)) then
19.     if (CheckTime(v, newX)) then
20.       \(bestX \leftarrow newX\) /* if constraints are met then save current solution */
21.       \(L \leftarrow L \cap d\) in bestX and remove request from \(L\) /*
22.     else
23.       \(T_r \leftarrow T_r \cup d\) /* if constraints not met, add request to tabu list \(T_r\) */
24.       \(newX \leftarrow bestX\)
25.   else
26.     \(T_r \leftarrow T_r \cup d\) /* if constraints not met, add request to tabu list \(T_r\) */
27.     \(newX \leftarrow bestX\)
28.   if (L.size == 0) then
29.     \(run \leftarrow false\) /* if no more requests left, then start next phase */
30. else
31.   \(T_v \leftarrow T_v \cup v\)
32.   \(T_r \leftarrow NULL\)
33.   \(newVehicle \leftarrow true\)
34. if (setV.size == T_v.size) then
35.   \(run \leftarrow false\) /* if no more vehicles are left then stop algorithm */
Algorithm 2: The overall structure for selecting vehicles and entering request

\[ \text{input: } \text{initial solution of phase 1 in } initX \]
\[ \text{output: } \text{A feasible vehicle routing solution for LSP trucks and taxis in } newX \]

```
while (run) do
  1. Initialize and fill list \( L \) // list of passenger request
  2. Initialize and fill list \( V \) // list of available taxis
  3. Initialize list \( T_r \) // empty tabu list of requests
  4. Initialize list \( C \) // cluster of passenger requests
  5. \( newX \leftarrow initX \) // decision variable \( x_{ij} \)
  6. \( bestX \leftarrow NULL \) // decision variable \( x_{ij} \)

  create clusters of request based on delivery location
  for (every cluster) do
    7. \( C \leftarrow \text{fill with requests in cluster} \) // based on delivery location
    8. \( w \leftarrow \text{closestDepot}(C) \)
    9. \( \text{newVehicle} \leftarrow true \)
    10. if (newVehicle) then
        11. \( v \leftarrow \text{selectVehicle.earliest}(w, V) \) // selects earliest taxi from depot \( w \)
        12. \( d \leftarrow \text{selectRequest.earliest}(i, L, T_r, C) \) // selects earliest passenger
        13. \( p \leftarrow \text{pickup location of } d \)
        14. \( newX \leftarrow \text{InsertRoute}(newX, v, i, p, d) \) // insert request in current vehicle route
        15. if (CheckCapacity(v, newX)) then
            16. if (CheckTime(v, newX)) then
                17. \( bestX \leftarrow newX \) // duplicate current solution into bestX
                18. \( L \leftarrow L \cap d \) // remove current request from list \( L \)
                19. \( \text{continue 8} \)
            19. else
                20. \( T_r \leftarrow T_r \cup d \) // add current request to tabu list \( T_r \)
                21. \( \text{continue 6} \)
        19. else
            20. \( T_r \leftarrow T_r \cup d \) // add current request to tabu list \( T_r \)
            21. \( \text{Continue 6} \)
        19. if (L.size == 0) then
            21. run \leftarrow false \ // no requests are left, proceed to phase 3
        20. else if (L.size == T_r.size) then
            21. \( v \leftarrow V \cup v \) // add vehicle to tabu list \( V \)
            22. \( T_r \leftarrow NULL \) // empty the request tabu list \( T_r \)
            23. \( \text{continue 6} \)
        19. else if (no vehicle left in \( V \) then)
            21. run \leftarrow false \ // if no vehicles left then stop algorithm
```
Appendix 5.3

Algorithm 3: The overall structure for selecting integrating scheduled lines for initial solution

**input**: Initial routing solution for LSP trucks and taxis in \( \text{initX} \) from the first two phases.

**output**: A feasible vehicle routing solution for the LSP trucks and taxis with the option to integrate scheduled lines

1. Initialize empty tabu list \( T^r \)
2. \( \text{oldCost} \leftarrow \text{cost}(\text{initX}) \) // saved the cost of the initial solution
3. \( \text{bestX} \leftarrow \text{initX} \) /* duplicate the solution of the phases into bestX and newX (decision variable \( x^r_{ij} \) */
4. let newZ be the scheduled lines decision variable
5. \( \text{newZ} \leftarrow \text{Initialize} \) // decision variable \( z^r_{ij} \)
6. For (all cargo requests) do
7. runF \( \leftarrow \text{true} \)
8. find \( \leftarrow \text{findOutlier(\text{newX}, T^r)} \) // find request with highest saving when removed
9. \( r \leftarrow \text{find.request} \) // save request in variable \( r \)
10. \( v \leftarrow \text{find.vehicle} \) // save vehicle in variable \( v \)
11. \( f \leftarrow \text{findSL}(r, T^r) \)
12. if \( (f == \text{NULL}) \) do
13. \( T^r \leftarrow T^r \cup \text{solution} \) // if no possible lines are found, add request to \( T^r \)
14. back to 7
15. while (runF) do
16. \( q \leftarrow \text{findBestTime}(f, r, \text{newX}, T^r) \) // find most suitable departure time of sched. line
17. if \( (q == \text{Null}) \) do
18. \( \text{newZ} \leftarrow \text{insertSchedLine(\text{newX}, \text{newZ}, f, r)} \) // insert the chosen repl. sched. line into route \( v \)
19. \( \text{newX} \leftarrow \text{insertPickupTaxi}(r, \text{newX}, \text{newZ}) \) /* add the request to a taxi route which picks up the request at end of repl. sched. line */
20. if (constraints are met) then
21. if \( (\text{cost(\text{newX}, \text{newZ})} < \text{oldCost}) \) then // if all constraints are met, compare cost
22. \( \text{bestX} \leftarrow \text{newX} \)
23. \( \text{bestZ} \leftarrow \text{newZ} \)
24. \( \text{runF} \leftarrow \text{false} \)
25. else
26. \( \text{newX} \leftarrow \text{bestX} \)
27. \( \text{newZ} \leftarrow \text{bestZ} \)
28. end if
29. \( T^r \leftarrow T^r \cup r \) // add request to tabu list \( T^r \)
30. else
31. \( T^r \leftarrow T^r \cup r \) // add request to tabu list \( T^r \)
Algorithm 4: Improvement phase method 1

input: Initial solution for PDP-SL
output: A feasible vehicle routing solution for the LSP trucks and taxis with the option to integrate scheduled lines

1. List C $\leftarrow$ create cluster off requests // based on the delivery locations
2. oldCost $\leftarrow$ cost(initX) // save the cost of the solution from phase 3
3. w $\leftarrow$ closestDepot(C) // depot closest to center of cluster
4. List V $\leftarrow$ find vehicles associated with w // list of vehicles of the selected depot
5. newX $\leftarrow$ initX // decision variable $x_{ij}^v$
6. bestX $\leftarrow$ initX // decision variable $x_{ij}^b$
7. newZ $\leftarrow$ initZ // decision variable $z_{ij}^r$
8. bestZ $\leftarrow$ initZ // decision variable $z_{ij}^b$

9. For (all requests in C) do

10. $f \leftarrow$ findSL(r, $y_{ri}$) // finds the most suitable repl. sched. line. for r
11. if ($f == NULL$) do
12. back to 9 // if no possible lines are found, select next request
13. newZ $\leftarrow$ insertScheduledLine(newX, newZ, f, v, r) // insert the first repl. sched. line
14. $f \leftarrow$ findSL(r, $y_{ri}$) // find the second repl. sched. line
15. if ($f == NULL$) do
16. Continue to 9 // if no possible lines are found, select next request
17. newZ $\leftarrow$ insertScheduledLine(newX, newZ, f, v, r) // insert the second repl. sched. line
18. newX $\leftarrow$ insertTaxi(r, newX, newZ, V) /* insert pickup at end transshipment location and delivery of request into a taxi route */
19. if (constraints are met) then
20. bestZ $\leftarrow$ newZ // save the current solution
21. bestX $\leftarrow$ newX
22. else
23. newZ $\leftarrow$ bestZ // reset to best feasible solution
24. newX $\leftarrow$ bestX
25. if (cost(bestX, bestZ) $>$ oldCost) then
26. bestZ $\leftarrow$ initZ // if the cost of solution is improved then save
27. bestX $\leftarrow$ initX
Appendix 5.5

Algorithm 5: Improvement phase method 2

**Input:** solution \textit{initX} and \textit{initZ} for PDP-SL

**Output:** improvement of the solution as \textit{bestX} and \textit{bestZ}

1. List \( C \) \( \leftarrow \) create list with outliers \hfill // outliers based on their delivery location
2. oldCost \( \leftarrow \) cost(initX) \hfill // save the cost of the initial solution
3. newX \( \leftarrow \) initX \hfill // decision variable \( x_{ij} \)
4. bestX \( \leftarrow \) initX \hfill // decision variable \( x_{ij} \)
5. newZ \( \leftarrow \) initZ \hfill // decision variable \( z_{ij}^{r_q} \)
6. bestZ \( \leftarrow \) initZ \hfill // decision variable \( z_{ij}^{r_q} \)
7. For (all requests in \( C \)) do
   8. fClosest \( \leftarrow \) findClosestSL(r) \hfill // chooses the closest repl. sched. line
6. newX \( \leftarrow \) insert(newX, newZ, fClosest, v, r)
9. While (run) do
   10. f \( \leftarrow \) findSL(C, \( v_i \)) \hfill // choose additional repl. sched. line
   11. if (f == NULL) do
   12. back to 7
   13. else if (f == fClosest)
   14. newZ \( \leftarrow \) insert(newX, newZ, f, v, r) \hfill // insert second repl. sched. line
   15. run \( \leftarrow \) false
   16. newX \( \leftarrow \) insertTaxi(r, newX, newZ, V) \hfill /* insert pickup at end transshipment location and delivery of request into a taxi route */
9. if (constraints are met) then
   10. if (cost(bestX, bestZ) < oldCost) then
   11. bestZ \( \leftarrow \) newZ \hfill /* if cost of solution is improved, save the new route
   12. bestX \( \leftarrow \) newX \hfill in decision variables bestX and bestZ */
   13. else
   14. newZ \( \leftarrow \) bestZ \hfill /* if cost of solution is not improved, reset newZ and
   15. newX \( \leftarrow \) bestX \hfill newX to last known best feasible solution */
   16. else
   17. newZ \( \leftarrow \) bestZ \hfill /* if constraints are violated, reset newZ and
   18. newX \( \leftarrow \) bestX \hfill newX to last known best feasible solution */
Algorithm 6: Improvement phase method 3

input: solution initX and initZ for PDP-SL
output: improvement of the solution as bestX and bestZ

1. oldCost ← cost(initX) // save the cost of the initial solution
2. newX ← initX // decision variable $x_{ij}$
3. bestX ← newX // decision variable $x_{ij}$
4. newZ ← initZ // decision variable $x_{ij}$
5. bestZ ← initZ // decision variable $x_{ij}$
6. list C ← create list // list with direct delivered requests
7. Sort.list(C) // sort requests in list C from most to least expensive
8. list I ← create all of the combinations /* combinations of binary system, indicates if request is tested in batch or not */
9. for (all batches in I) do
   10. for (all requests r in current batch) do
      11. if (r == NULL) do end improvement method
      12. newX ← findSL(newX, r) // finds most suitable repl. sched. line
      13. newZ ← insertSL(newZ, r, f, q) // insert repl. sched. line $f$
      14. newX ← insertTaxi(newX, newZ, r) /* insert pickup at end transshipment location and delivery of request into a taxi route */
      15. if (constraints are met) then
      16. bestX ← newX // save the new solution
      17. newX ← bestX // reset to last known best feasible solution
      18. back to 11
      19. if (cost(newX, newZ) < oldCost) then
      20. remove current batch from list C /* remove requests from the list if the cost of the solution has decreased */
      21. back to 6
      22. else
      23. newX ← bestX /* if the cost has not decreased than reset to last known feasible solution */
      24. newZ ← bestZ
      25. back to 10
## Appendix 6.1

### Table 9 Tuesday dataset results with 15 taxi passengers

| Sl    | Truck cost | Taxi cost | SL dist. | Truck dist. | Taxi dist. | SL price | Taxi price | SL used | SL cargo | SL cost | LSP cost | LSP benefit | Taxi benefit | Society benefit | Heuristic duration |
|-------|------------|-----------|----------|------------|------------|----------|------------|---------|----------|---------|----------|-------------|---------------|-----------------|-------------------|--------------------|
|       | (€)        | (€)       | (km)     | (€)        | (€)        | (€)      | (€)        | (#)     | (#)      | (#)     | (€)      | (€)         | (€)           | (€)            | (min)              |
| 0.00  | 0.80       | 0.30      | 66,90    | 53,52      | 125.00     | 37.50    | 15.63      | 32.70   | 26.16    | 171.50  | 51.45    | 21.44       | 13.00         | 11.75          | 0.00               | 40.11              |
| 0.00  | 0.80       | 0.30      | 66,90    | 53,52      | 125.00     | 37.50    | 15.63      | 32.70   | 26.16    | 171.50  | 58.60    | 21.44       | 13.00         | 11.75          | 0.00               | 44.76              |
| 0.00  | 0.90       | 0.30      | 66,90    | 60,21      | 125.00     | 37.50    | 15.63      | 31.10   | 27.99    | 180.30  | 54.09    | 22.54       | 14.00         | 12.63          | 0.00               | 44.58              |
| 0.00  | 0.90       | 0.30      | 66,90    | 60,21      | 125.00     | 37.50    | 15.63      | 32.70   | 29.43    | 171.50  | 68.60    | 21.44       | 13.00         | 11.75          | 0.00               | 48.03              |
| 0.50  | 0.80       | 0.30      | 66,90    | 53,52      | 125.00     | 37.50    | 15.63      | 32.80   | 26.24    | 168.90  | 50.67    | 21.11       | 12.00         | 10.75          | 5.38              | 44.79              |
| 0.50  | 0.80       | 0.40      | 66,90    | 53,52      | 125.00     | 37.50    | 15.63      | 46.90   | 37.52    | 136.25  | 54.50    | 17.03       | 8.00          | 6.75           | 3.38              | 45.40              |
| 0.50  | 0.90       | 0.30      | 66,90    | 60,21      | 125.00     | 37.50    | 15.63      | 33.80   | 30.42    | 166.90  | 50.07    | 20.86       | 12.00         | 10.75          | 5.38              | 48.37              |
| 0.50  | 0.90       | 0.40      | 66,90    | 60,21      | 125.00     | 37.50    | 15.63      | 36.60   | 32.94    | 163.20  | 65.28    | 20.40       | 10.00         | 8.75           | 4.38              | 52.60              |
| 1.00  | 0.80       | 0.30      | 66,90    | 53,52      | 125.00     | 37.50    | 15.63      | 36.30   | 29.04    | 163.20  | 48.96    | 20.40       | 10.00         | 8.75           | 8.75              | 49.25              |
| 1.00  | 0.80       | 0.40      | 66,90    | 53,52      | 125.00     | 50.00    | 15.63      | 46.90   | 37.52    | 136.25  | 54.50    | 17.03       | 8.00          | 6.75           | 6.75              | 48.77              |
| 1.00  | 0.90       | 0.30      | 66,90    | 60,21      | 125.00     | 37.50    | 15.63      | 32.67   | 29.40    | 163.20  | 48.96    | 20.40       | 10.00         | 8.75           | 8.75              | 49.61              |
| 1.00  | 0.90       | 0.40      | 66,90    | 60,21      | 125.00     | 50.00    | 15.63      | 46.90   | 42.21    | 136.25  | 54.50    | 17.03       | 8.00          | 6.75           | 6.75              | 53.46              |
| 2.00  | 0.80       | 0.30      | 66,90    | 53,52      | 125.00     | 37.50    | 15.63      | 46.90   | 37.52    | 136.25  | 40.88    | 17.03       | 8.00          | 6.75           | 13.50             | 54.40              |
| 2.00  | 0.80       | 0.40      | 66,90    | 53,52      | 125.00     | 50.00    | 15.63      | 46.90   | 37.52    | 136.25  | 54.50    | 17.03       | 8.00          | 6.75           | 13.50             | 55.52              |
| 2.00  | 0.90       | 0.30      | 66,90    | 60,21      | 125.00     | 37.50    | 15.63      | 46.90   | 42.21    | 136.25  | 54.50    | 17.03       | 8.00          | 6.75           | 13.50             | 59.09              |
| 2.00  | 0.90       | 0.40      | 66,90    | 60,21      | 125.00     | 50.00    | 15.63      | 46.90   | 42.21    | 136.25  | 54.50    | 17.03       | 8.00          | 6.75           | 13.50             | 60.21              |
| 3.00  | 0.80       | 0.30      | 66,90    | 53,52      | 125.00     | 37.50    | 15.63      | 66.90   | 53.52    | 125.00  | 50.00    | 15.63       | 0.00          | 0.00           | 0.00              | 53.52              |
| 3.00  | 0.80       | 0.40      | 66,90    | 53,52      | 125.00     | 50.00    | 15.63      | 66.90   | 53.52    | 125.00  | 50.00    | 15.63       | 0.00          | 0.00           | 0.00              | 53.52              |
| 3.00  | 0.90       | 0.30      | 66,90    | 60,21      | 125.00     | 37.50    | 15.63      | 66.90   | 60.21    | 125.00  | 50.00    | 15.63       | 0.00          | 0.00           | 0.00              | 60.21              |
| 3.00  | 0.90       | 0.40      | 66,90    | 60,21      | 125.00     | 50.00    | 15.63      | 66.90   | 60.21    | 125.00  | 50.00    | 15.63       | 0.00          | 0.00           | 0.00              | 60.21              |

### Notes
- Includes Scheduled Lines (SL)
- Benefit (€) includes the cost of taxies.
### Appendix 6.2

**Table 10 Tuesday dataset results with 45 taxi passengers**

| No Scheduled Lines (SL) | Includes Scheduled Lines (SL) | Benefit | Society Benefit | Heuristic | SL cost | Taxi SL cost | LSP cost | LSP benefit | Taxi benefit | Society benefit | Social benefit | LSP cost benefit | Taxi cost benefit | Heuristic duration (min) |
|------------------------|-------------------------------|---------|-----------------|-----------|---------|-------------|-----------|-------------|--------------|-----------------|-----------------|----------------|---------------------|-----------------------|--------------------------|
|                        |                               |         |                 |           |         |             |           |             |              |                 |                 |                |                     |                       |                          |
|                        |                               |         |                 |           |         |             |           |             |              |                 |                 |                |                     |                       |                          |

### Appendix 6.3

**Table 11 Wednesday dataset results with 15 taxi passengers**

| No Scheduled Lines (SL) | Includes Scheduled Lines (SL) | Benefit | Society Benefit | Heuristic | SL cost | Taxi SL cost | LSP cost | LSP benefit | Taxi benefit | Society benefit | Social benefit | LSP cost benefit | Taxi cost benefit | Heuristic duration (min) |
|------------------------|-------------------------------|---------|-----------------|-----------|---------|-------------|-----------|-------------|--------------|-----------------|-----------------|----------------|---------------------|-----------------------|--------------------------|
|                        |                               |         |                 |           |         |             |           |             |              |                 |                 |                |                     |                       |                          |
|                        |                               |         |                 |           |         |             |           |             |              |                 |                 |                |                     |                       |                          |
|                        |                               |         |                 |           |         |             |           |             |              |                 |                 |                |                     |                       |                          |
|                        |                               |         |                 |           |         |             |           |             |              |                 |                 |                |                     |                       |                          |

---

**SL:** Service Level

**Taxi SL cost:** Cost for taxi services

**LSP cost:** Cost for LSP services

**Benefit:** Benefit for the society

**Society Benefit:** Benefit for the society

**Heuristic duration:** Duration of the heuristic algorithm
## Appendix 6.4

### Table 12 Wednesday dataset results with 45 taxi passengers

<table>
<thead>
<tr>
<th>SL</th>
<th>Truck cost</th>
<th>Taxi cost</th>
<th>Distance (km)</th>
<th>Truck distance</th>
<th>Taxi distance</th>
<th>Taxi price</th>
<th>SL used</th>
<th>SL cargo</th>
<th>SL cost</th>
<th>LSP cost</th>
<th>Benefit</th>
<th>Benefit</th>
<th>Benefit</th>
<th>Heuristic Duration (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.80</td>
<td>0.30</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>29.30</td>
<td>23.44</td>
<td>703.10</td>
<td>210.93</td>
<td>87.89</td>
<td>38.00</td>
<td>28.73</td>
</tr>
<tr>
<td>0.00</td>
<td>0.80</td>
<td>0.40</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>29.30</td>
<td>23.44</td>
<td>703.10</td>
<td>281.24</td>
<td>87.89</td>
<td>38.00</td>
<td>28.73</td>
</tr>
<tr>
<td>0.00</td>
<td>0.90</td>
<td>0.30</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>29.30</td>
<td>26.37</td>
<td>703.10</td>
<td>210.93</td>
<td>87.89</td>
<td>38.00</td>
<td>28.73</td>
</tr>
<tr>
<td>0.00</td>
<td>0.90</td>
<td>0.40</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>29.30</td>
<td>26.37</td>
<td>703.10</td>
<td>281.24</td>
<td>87.89</td>
<td>38.00</td>
<td>28.73</td>
</tr>
<tr>
<td>0.50</td>
<td>0.80</td>
<td>0.30</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>29.30</td>
<td>23.44</td>
<td>703.10</td>
<td>210.93</td>
<td>87.89</td>
<td>38.00</td>
<td>28.73</td>
</tr>
<tr>
<td>0.50</td>
<td>0.80</td>
<td>0.40</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>29.30</td>
<td>23.44</td>
<td>703.10</td>
<td>281.24</td>
<td>87.89</td>
<td>38.00</td>
<td>28.73</td>
</tr>
<tr>
<td>0.50</td>
<td>0.90</td>
<td>0.30</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>29.30</td>
<td>26.37</td>
<td>703.10</td>
<td>210.93</td>
<td>87.89</td>
<td>38.00</td>
<td>28.73</td>
</tr>
<tr>
<td>0.50</td>
<td>0.90</td>
<td>0.40</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>29.30</td>
<td>26.37</td>
<td>703.10</td>
<td>281.24</td>
<td>87.89</td>
<td>38.00</td>
<td>28.73</td>
</tr>
<tr>
<td>1.00</td>
<td>0.80</td>
<td>0.30</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>29.30</td>
<td>23.44</td>
<td>705.10</td>
<td>211.53</td>
<td>88.14</td>
<td>36.00</td>
<td>27.23</td>
</tr>
<tr>
<td>1.00</td>
<td>0.80</td>
<td>0.40</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>29.30</td>
<td>23.44</td>
<td>705.10</td>
<td>282.04</td>
<td>88.14</td>
<td>36.00</td>
<td>27.23</td>
</tr>
<tr>
<td>1.00</td>
<td>0.90</td>
<td>0.30</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>29.30</td>
<td>26.37</td>
<td>705.10</td>
<td>282.04</td>
<td>88.14</td>
<td>36.00</td>
<td>27.23</td>
</tr>
<tr>
<td>1.00</td>
<td>0.90</td>
<td>0.40</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>29.30</td>
<td>26.37</td>
<td>705.10</td>
<td>282.04</td>
<td>88.14</td>
<td>36.00</td>
<td>27.23</td>
</tr>
<tr>
<td>2.00</td>
<td>0.80</td>
<td>0.30</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>2.00</td>
<td>0.80</td>
<td>0.40</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>77.00</td>
<td>61.60</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>2.00</td>
<td>0.90</td>
<td>0.30</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>201.45</td>
<td>83.94</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>2.00</td>
<td>0.90</td>
<td>0.40</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>77.00</td>
<td>69.30</td>
<td>671.50</td>
<td>268.60</td>
<td>83.94</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>