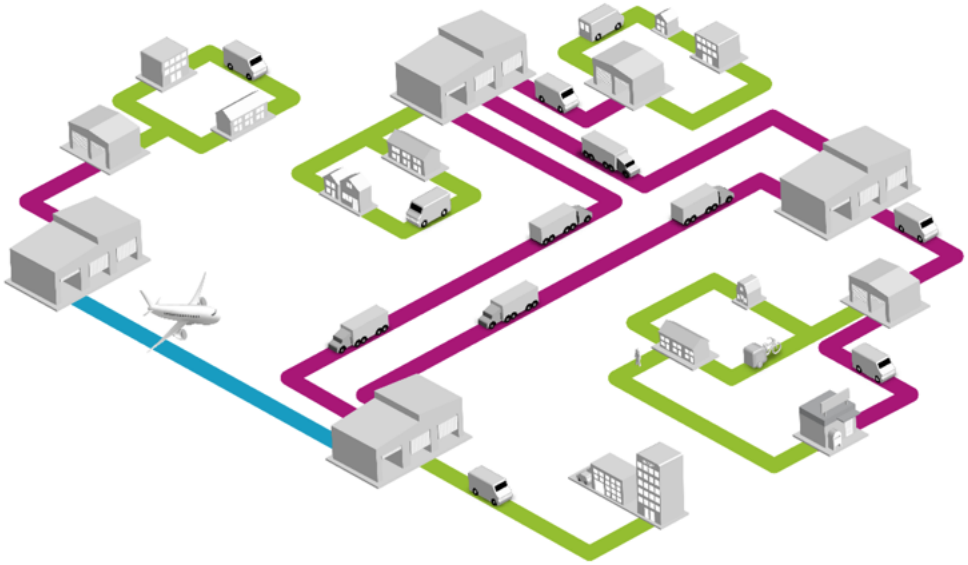
Profit Optimization in Express Networks

Casper van Dijk

Master of Science Thesis



TU Delft Delft University of Technology

Department of Applied Mathematics

Profit Optimization in Express Networks

Master of Science Thesis

For the degree of Master of Science in Applied Mathematics at Delft University of Technology

Specialization: Optimization

Casper van Dijk
Student number: 4261577

12 September 2018

Thesis committee:
Dr. ir. K.I. Aardal (TU Delft)
A. Tossenberger MSc. (ORTEC)
M. Wingender MSc. (ORTEC)
Dr. ir. G.F. Nane (TU Delft)

Faculty of Electrical Engineering, Mathematics and Computer Science (EEMCS)
Delft University of Technology

The work in this thesis was supported by ORTEC. Their cooperation is hereby gratefully acknowledged.

Abstract

This report proposes a solution approach for the problem of maximizing the profit of a parcel delivery company. The profit of a company depends on the market share that it has for each of its commodities. In turn, the market shares depend on the price and service time that the company offers the customers in comparison with the offer of competitors in the market. The classical approach to maximizing the profit, is to first determine the prices, which in turn determine the demand, and then to minimize the costs in the network, part of which is the routing of the parcels. The solution approach that is proposed integrates this process, which improves the quality of the solutions that are found, and the amount of time that is needed to find them. Part of the contribution of the project is the proposal of a new meta-heuristic algorithm that can also be used on similar problems, and which is based on the principle of Local Branching. Furthermore, since parts of the solution approach are applicable in a broader context, the contribution also lies in the insight that the report provides into other network optimization models. I.e., it shows the impact of multiple aspects on the solvability of these models, both in the area of problem definition and input data. Moreover, the combination of optimizing the prices and routing for multiple commodities, while also optimizing network decisions in a hub-and-spoke network, is not yet covered in the literature.

The proposed optimization approach is split into two parts based on the difficulty of the situation. In Situation 1, only the pricing and routing is optimized. In Situation 2, the same is done, but there is also more freedom for optimizing the network. For Situation 1, the approach uses a linearization of the objective function to approximate the originally non-linear model by a linear one. For this model, an efficient model formulation and heuristic methods are proposed, which were tested for the expected effect on the solution time and error. A general purpose MILP solver is used to find a solution. For Situation 2, the proposed approach uses the approach from Situation 1 as a basis, and extends this with a meta-heuristic algorithm, based on Variable Neighborhood Search and Local Branching. It uses a general purpose MILP solver to search the neighborhood of the current solution for better solutions. The parameters of the algorithm are automatically determined by using problem-specific knowledge.
Furthermore, the findings of the two situations were applied to a realistic large-network case. Here we concluded that the best approach in complex situations like this, is to use the MILP

solver to find a reasonably good starting solution, and to use the proposed algorithm to improve upon that solution.

We expect that for Situation 1 and networks that have at most 100 depots, 4 hubs, and 5000 commodities, an error between 1% and 2% is possible in a few hours of solution time on a standard machine. For these networks and Situation 2, the error is higher, harder to predict, and a guarantee on the quality is not possible. This is because the proposed algorithm is used for these situations, but we nonetheless expect an error of at most 3% is possible within a few hours of solution time.
Another important conclusion we have drawn, is that introducing the setting of prices into the problem (instead of only minimizing the costs in the network) increases the complexity of the problem considerably. When the prices are chosen from a continuous range, the solution time is expected to increase by more than a factor of 50. Furthermore, the analysis of the impact of the input data on the solution time, shows that from the different kinds of input data, the hub capacities have by far the greatest impact on the solution time. We also concluded that relaxing the constraints that ensure one and only one path is chosen per commodity is very useful, as it has very little impact on the solutions that are found, but *does* decrease the solution time.
A more general conclusion we have drawn, is that because MILP solvers are black-box algorithms, discovering their workings can only be done by changing the input and analyzing the output. Although this gives great insight for many general aspects, investigating the more detailed aspects often leads to unpredictable results. An example of this is the formulation of constraints, for which we have shown that a stronger formulation does not necessarily lead to a shorter solution time.

Preface

After a fairly long and interesting study path, starting at Architectural Engineering, and via Maritime Technology, my studies will now end at the faculty of Applied Mathematics. During these last years, I developed a fondness for Optimization, and as ORTEC is one of the major companies in optimization in the Netherlands, I am very grateful that I was able to finish my master's degree with this company. I am also very happy that I have been given the chance to start my career here, and I can't wait to finally put all that acquired knowledge into practice.

As for the project itself, it came into being after a customer of ORTEC had a question similar to the research question of this thesis. Because ORTEC had never answered a question like this, and the time that was available for it was limited, the implementation was suboptimal. To get a better view of the (im)possibilities in approaching a project like this, a vacancy for a graduation thesis was published, of which the result is the report lying in front of you.

I would like to acknowledge everyone that was somehow involved in the work that I did for this thesis, I learned a lot, and for that, I thank you. Special thanks to my daily supervisor Anna Tossenberger, who, whenever it was needed, made time for me and gave me valuable feedback. I really enjoyed having all those mathematical and business discussions. Also many thanks to the supervising professor, Karen Aardal, for the friendly and insightful meetings. Next, my thanks go out to Marc Wingender, who guided the process from the ORTEC side, and also provided sharp feedback. And finally, I also want to thank Robin for the moral support, which at times was much needed and wonderful.

I hope you will enjoy reading the report, and that you will find that you have gained value from it.

Delft, University of Technology — Casper van Dijk
12 September 2018

This thesis is dedicated to my parents for their unwavering support.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter covers an introduction of the subject matter and the project.

1-1 Context of the project and motivations

1-1-1 The subject

Parcel delivery companies offer time-guaranteed transportation of parcels, letters, and packages of different sizes, picked up at one customer and delivered off at another. From this point on, we will use the word parcel for any of these three types. The time at which the parcel is guaranteed to arrive depends on the service, which defines the time by which the parcel has to arrive at its destination, that the customer is paying for. To transport the parcels, a network consisting of facilities and links is used. In general, the amount of parcels sent between the different areas does not justify a fully connected network, which is why one or more intermediate collection and processing points are used. Networks of this type are said to have a hybrid (because some connections are direct) Hub and Spoke (HS)-topology, and significantly reduce the number of links required to connect all origin- and destination-areas.

1-1-2 The project

The *market share*, defined as the percentage of all units in the market that are transported by the company, depends on the price and service time the company offers the customers in comparison with the offer of competitors in the market. If the company has relatively high prices, the market share will be low, but the parcels that are transported result in a higher revenue per parcel. If prices are lower, the market share is higher which will result in higher volumes for transport, which generally results in more (cost) efficient transport. But in this case, the revenue per parcel is also lower. Companies determine for the most part themselves what their prices are (the *pricing strategy*). Something similar holds for the service time. If a competitor can transport a parcel quicker and for the same price, the market share will

decrease. However, quicker transport usually results in higher cost levels. Optimizing the *total profit*, the total revenue minus the total cost, is an important objective for these companies. The classical approach to this is first to determine what the demand is for the different services that could be offered, then to determine what services the company actually wants to offer, and in a last step to optimize the network that will be used to accommodate this demand. This project is aimed at finding an approach that integrates these steps, and in this way finding better quality solutions. The complex set of relationships between prices, levels of service, network design, and efficiency is the subject of this project. The main difference with standard multi-commodity flow problems is the possibility of influencing the flow quantities.

1-1-3 Motivation

For express service providers there is an ever-increasing demand for the transshipment of parcels of different sizes. The market is divided into two main areas. B2B refers to the business-to-business segment, which is historically the main segment in which express service providers operate. And B2C, which refers to the business-to-consumer segment, which is an upcoming market for these companies. They operate in the so called Courier, Express and Parcel (CEP) market, which according to figures of A.T. Kearney (Salehi et al. (2015)) was worth €43.1 billion in 2013 in Europe alone, and was expected to grow steadily. At the same time the *profit margins*, the ratio of profit to revenue, of all highly networked businesses have been squeezed as a result of intensive price competition. To stay competitive in this market, network theory, in the broadest sense of the word, has mostly been used for the minimization of costs. This is reflected in the research that has been published where not much attention has been given to profit optimization (O'Kelly et al. (2015)). A reason for this is that in most situations, information regarding the relation between price, market share, and service time is not available or lacking in quality. In some situations however, this information is available, and then finding out what the optimal pricing strategy is can prove very beneficial. On top of that, as computer power and algorithms advance, more and more aspects of the business of express service providers will be brought together into one problem. The step of including the pricing strategy in the network optimization is an example of this.
Furthermore, the results of this thesis are not only applicable to situations in the CEP market, since the hub and spoke structure is also used in networks dealing with information or people. And thus other network based businesses that require price setting (e.g. airlines, freight transporters, electricity suppliers) can also benefit from an approach that integrates price setting and network optimization.
The contributions of the thesis are covered in the literature review in Section 1-4.

1-2 The express supply chain and definitions

What follows is a description of the daily operations of an express service provider. Definitions that are useful in the rest of the report are in *italic*.
Express service providers (or express service carriers) collect parcels from customers, which

can be persons or companies, at their front door. From there they are transported to a local sorting center (or *depot*) by small trucks going on *pickup paths*. The delivery of parcels is also done from a depot, and it is either combined with a pickup path or it is done via a dedicated *delivery path*. This whole process is called *the Pick Up and Delivery (PUD)-process*. The transportation from an *origin depot* (the depot the parcel is brought to after it is picked up) to a *destination depot* (the last facility that a parcel visits before delivery) is usually not done directly but through one or more *hubs*, larger facilities that collate and process parcels coming from multiple depots or other hubs. While hubs and detours lead to extra costs, this is compensated through the effects of parcel consolidation (e.g. economies of scale). The *service time* is defined as the time by which the parcel is guaranteed to be delivered at the customer (e.g. by 9 AM the next morning, or by 5 PM two days after sending the parcel). The combination of an origin depot, a destination depot, and a service is called a commodity. Every commodity has a *path*, which defines the hubs that the parcels visit. For every depot-to-hub-, and hub-to-hub-connection, transportation is done by road or air depending on the amount of parcels and the distance that needs to be covered. The part of the process between the origin and destination depot is called the *line-haul* process. The separation between the PUD-process and the line-haul process is called a *pickup or delivery cut-off time*, a time by which the parcels must be available at the depot for sorting and further transportation. An overview of the process is depicted in Figure 1-1.
Furthermore, below are some additional and useful definitions regarding different parts of the transportation process:

Location: A (potential) depot or hub.

Movement: The part of a tour between two consecutive locations.

Empty movement: The relocation of a vehicle during which no parcels are transported.

Arc: The link between two locations in the network.

Flow: A quantity of parcels transported between two points in the network.

Fixed/investment costs: Costs that are independent of the amount of parcels that are transported, examples are construction costs for depots, and maintenance costs for trucks.

Variable/operational costs: Costs that are dependent on the flow of parcels that is transported. These costs have two components; costs based on the weight of the parcels and on the amount of parcels. Examples are the handling costs of hubs, and the part of the costs for trucks that are dependent on weight.

Line-haul costs: These consist of all costs of transporting the parcels through the network. Costs for the PUD-process are not included in this.

1-2-1 Network optimization

The way in which the network is designed has a big influence on the total profit of the company that operates it. Decisions that have to be made in this regard can be divided into the following categories:

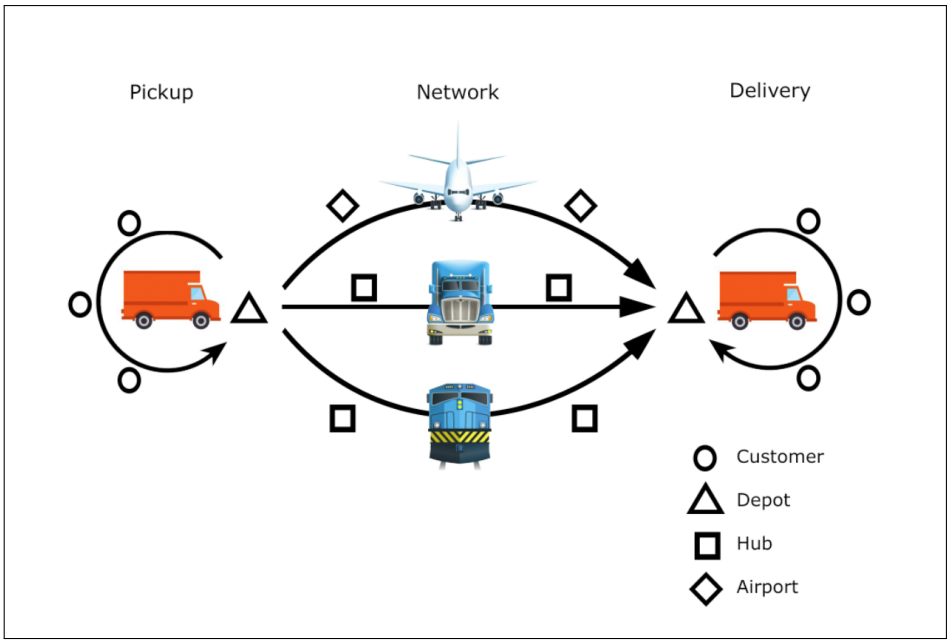


Figure 1-1: Schematic overview of the delivery process

Areas A group of customers in a certain area, e.g. corresponding to a postal code, need to be assigned to a depot from which the pickup and delivery paths are done. It is also an option to decide not to include an area because transporting the parcels from and to it is not profitable. These decisions influence the quantity of flow and its direction in the network.

Depots The amount and location of the depots are other network design variables. The location influences the distance and time to customers. As for the amount of depots, having many means the average distance and time to customers is relatively small which results in low PUD costs. But it also means more (fixed) depot costs for building, land use and equipment. For deciding on fewer depots the opposite holds.

Hubs Of all the network design variables, the decision on the amount and location of the hubs has the greatest direct influence on the total profit. They have a central and key role in what paths the parcels can take through the network, and thus which costs are to be made. Also, they represent the biggest fixed costs of all parts of the network. Having many hubs means short inter-hub and depot-hub distance, and less detours through other hubs, and thus lower line-haul costs. It also means more hub touches for the parcels, and so higher variable hub costs. And finally the fixed hub costs are relatively high. Having few hubs means the exact opposite of this.

Arcs In practice, the decision on the arcs entails the planning of transportation between two facilities (depots and/or hubs). This is represented by fixed costs, and opening an arc means there are more paths that the parcels can take.

Paths The decisions on the arcs and hubs determine the paths that a parcel can take through the network. Every path then has a variable costs and from these paths the one is selected that results in the highest total margin.

An elementary example on how the different decisions influence one another is given in Figure 1-2. In this situation we are looking at sending a flow of parcels from area A1 to area A6. The fastest (and probably the cheapest when only looking at the variable costs) path is A1-D1-D2-A6. This would mean the arc D1-D2 and the depot D2 would have to be opened, and A6 would have to be assigned to depot D2. Another option is choosing the path A1-D1-H2-H3-D4-A6. This would result in higher line-haul costs since more distance is covered and more hubs are visited, and higher PUD costs since the distance D4-A6 is high, but no new facilities would need to be opened. Yet another option is A1-D1-H1-D2-A6. For this, hub H1 would need to be opened, but it can then also be used by parcels having different origins or destinations such as D2-D3, and result in lower total costs. A final option is deciding not to include area A6 at all. This would save a lot of fixed and variable costs, but would also mean the revenue from parcels being sent from or to A4, A5, and A6 are excluded.

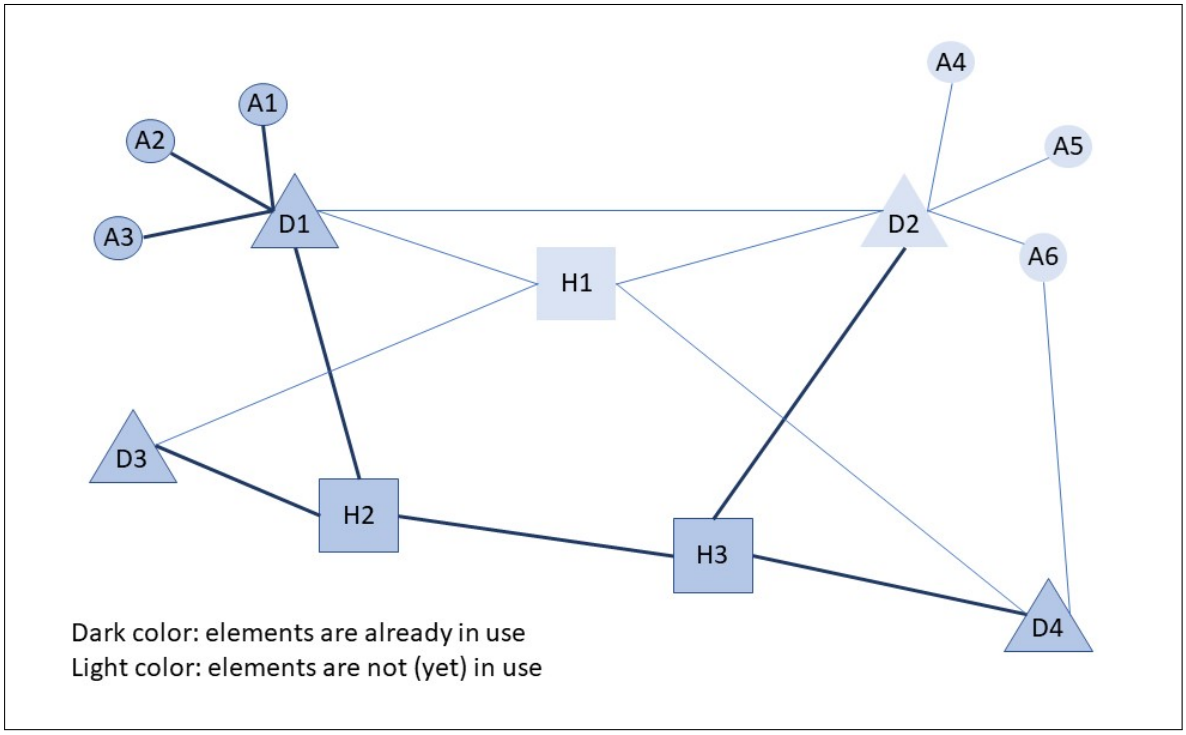


Figure 1-2: Influence of different network design variables

These are the relevant network design considerations in this thesis. The main element that is added to this, is the regulation of the flow by determining the pricing strategy. This is further explained in the next section.

1-3 Problem description

The core of the problem that we're trying to solve is the setting of prices for the different commodities in an express network, which in turn determines the flow in the network. This translation from prices to flow is done using a relation between price and market share. The aim of the project is to find a solution approach for maximizing the total profit. The approach should be a basis for further development into a more detailed and realistic model that can be used to make strategic decisions. Studying the key components, assumptions, and limitations of the approach are therefore an important part of the project aim. Even though the approach will only be used for strategic purposes, which does not require a short solution time, it is of paramount importance that the time it takes to execute the solution approach is limited as much as possible. This is because it should be possible to use the approach on large networks, and to add details that complicate the problem, without making the solution time too long for the approach to be usable. There are two main situations that are covered in the project; in the first situation a company has a fully functional network in place with only minor room for optimization of the network, and in the second situation more of the network is available for optimization.

Research question Which solution approach is most suitable for finding the optimal pricing strategy and network structure in Situations 1and 2, when maximizing the total profit is the objective? And which conclusions can we draw regarding the computational complexity?

1: "The Fixed Network" The number of hubs and their locations are fixed and they have a fixed capacity, depots have a fixed location and cannot be closed, and arcs are allowed to be opened or closed.

2: "The Variable Network" The number of hubs, their locations, and their capacity need to be chosen. Depots have a fixed location, and both the depots and arcs are allowed to be opened or closed.

An other objective of the thesis is gaining insight into the factors that influence similar problems.

1-3-1 Scope

Choices have to be made about what aspects of actual express network operations to include or exclude in the approach. Since we are interested in a general solution approach for strategic purposes, the main criteria for including or excluding an aspect are: The importance of an aspect in actual operations, the difficulty of accurately modeling it, the consequences for the applicability of the solution approach to other networks, and the impact the aspect has on the time that is needed to find a good quality solution.

One of the aspects that are excluded is dynamic behavior, by which we mean the change of variables over time. An example of this is the gradual decrease of the market share if a price is increased. What we are investigating in this project is an economic equilibrium in the market, an end-state after all internal and external influences have remained unchanged

for an unlimited lime. For us this means that dynamic behavior due to changes made by the company itself or its competitors is not in the scope of this project.
We will also not look at the decisions to include/exclude an area or assign it to a depot. One reason for this is that in practice the information necessary to include this, such as the total flow for every pair of areas, will not be available in practice. Furthermore, the amount of variables in the problem would dramatically increase, and this would make dealing with this aspect the focus of the project, and we think the combination of other possible aspects would give the project a more relevant contribution.
Next, some "smaller" aspects are presented that are out of scope of the project:

- Interdependence between OD-pairs
- Fluctuations in demand (average day in average week)
- Stochastic demand
- Daily timing (the decision on movements, tours, drivers, fleet size)
- Penalties on violated service times
- Different costs for different types of parcels
- Path generation
- Multiple vehicle types (specifics of air networks are also out of scope)
- Dedicated freight hubs

An aspect that is *in* the scope of the project is the inclusion of multiple service levels. This means that for every Origin-Destination (OD)-pair parcels can be delivered in one, two, or three days. A useful application of this is that it can be used to determine the best set of services to offer.
Vehicle balancing is also included in the problem, which means that the costs for repositioning vehicles because of a flow imbalance are taken into account. This imbalance occurs when the sum of the market share times the total flow taken over all commodities that have a certain depot as the origin, is different than the same sum but over all commodities that have that depot as the destination.

1-3-2 Assumptions

For this project we assume that for every commodity a deterministic relation is known between price and market share which models customer behavior in deciding which company to use for sending parcels in a competitive market. Also, every commodity is a sub-market not influenced by other sub-markets, so the market share of a commodity is not dependent on the price or service level of other services or OD-pairs. In reality this information is better represented by a stochastic relation dependent on more than only the commodity's own price. But since in practice this information will very likely not be available, and we are interested in strategic (long-term) and tactical (medium-term) decisions, the simple deterministic relation suffices for this project.

We assume that the function that determines the market share as a function of the price has a few characteristics. The reason for some of these assumptions is covered in Subsection 3-4-1.

- The range includes 0: it must be possible to set the market share to zero by setting the price to its maximum.
- The function is non-increasing: a higher price results in a lower market share.
- The function is a concave polynomial.

We also assume that the function will only be defined on a limited domain, since determining a function that reflects reality well will not be practically possible on a large domain. Furthermore, the timescale at which we are looking is one day, so all costs and demands must be given or calculated for a regular day in a regular week.
Some other information we assume to be given:

- Total flow in the market per commodity.
- The precise definition of the service levels that can be offered.
- Hub capacities.
- Cost parameters for arcs, depots, hubs.
- Transportation characteristics and timing information on depots and hubs for the generation of paths.

1-3-3 Outline of the approach

Methods in the field of mathematical programming lend themselves very well to solve problems like the one stated in this thesis. This is why the problem will be modeled as an *optimization model*, and using methods from the field of mathematical programming will be hugely important in finding an optimal approach. For an overview of the mathematical background of the approach, we refer the reader to Chapter 2.
All computational work on the models is done in AIMMS, this is a software system designed for modeling and solving large-scale optimization and scheduling-type problems (Kallrath (2004)). Background on this system is covered in Section 2-3.

1-3-4 Challenges

In answering the research question, there are a few challenges that are worth discussing upfront. Every one of these challenges is also covered in 1-4, 3-3 or 4-3, where a short overview of the approach followed in previous research in dealing with these challenges is given.

- Because of the amount of variables and constraints, and the complexity of the model, we cannot expect that solving it to optimality is possible in a time span reasonable for using the model in a decision-making context. This means a solution approach will need to be found that either drastically improves the time in which the solution is found for a specific instance of the problem, or that approximates the solution to within some known range.
- The way in which the costs should be modeled, and to what level of detail is another challenge. The choices made here have both a great influence on the correctness of the model when compared to reality, and the mathematical complexity. Considerations

in this area will usually be a trade-off between these two. An example is the cost of transportation between hubs, which can be modeled as a step function, reflecting reality where trucks or planes are used, or as a continuous linear function, which will make the model easier to solve but is less accurate.

- When trying to find the optimum of a mathematical programming model, non-linearity of the objective function generally makes it a lot harder to get good results. This non-linearity is an inherent property of the problem we are trying to solve, since the profit is a product of the flow of parcels and the price that is asked for them, where the flow is dependent on the price.
- When optimizing the structure of the network, the path that every parcel takes is either fixed or variable. When the latter approach is chosen, the complexity of the model will very quickly increase since in any realistically sized network there are lots of potential paths for every parcel. So an approach must be found that takes this into account.
- We are dealing with price and market share, so consumer choice needs to be taken into consideration because that determines the amount of flow in response to a certain price setting. This is a complicated subject, especially if we assume a customer can choose to send a parcel using different competing service levels. This has a great effect on the usefulness of the outcome, and so careful assumptions need to be made.
- The most important computational difficulty of conventional network-design formulations is the weakness of their Linear Programming (LP)-relaxations. This is a problem when applying some form of Branch And Bound (B&B)-method, because for these methods the strength of the LP-relaxations have a great effect on the speed with which the found solutions converge to the optimum. More on LP-relaxations and B&B is covered in Section 2-1.

1-4 Literature

This literature overview is given with two aims. The first is showing the relation to other research, so the differences and similarities are presented, along with the contributions of this thesis. The second is analyzing the approach followed by other researchers when faced with the challenges that are presented in Section 1-3. Not everything is covered in the literature Section of this chapter, for topics that are specific to the individual situations of the research question, we refer the reader to Sections 3-3 and 4-3. The first of the aforementioned goals is covered in the sections called *Context*, where the focus is on general problem descriptions. The second is covered in the sections called *Challenges*, which have more attention for specific approaches to the challenges.

1-4-1 Context

The problem considered in this thesis is based on network flow theory, and all network flow concepts are rooted in graph theory and graph notation. Practically all theory of network flows is based on Linear Programming and is closely related to the field of Operations Re-

search. Network-flow theory is one of the best studied and developed fields of optimization. Besides "standard" applications in transportation, scheduling, etc., it has relations to quite different fields of science and technology such as combinatorial mathematics, algebraic topology, electric circuit theory, geographic information systems, etc. (Iri (1996)). Also, computer scientists have given significant attention to the efficient implementation of methods used to find solutions to these problems.
Within the field of network flow theory, we are most interested in the part that concerns multi-commodity flow problems, in which every commodity has an origin, a destination and a demand. They are regarded among some of the most difficult Mixed Integer Linear Programming (MILP) problems (Castro (2003)).
A further specialization that is relevant for our approach is the setting of prices for the different commodities, where the demand is dependent on the price. This is not an extensively studied subject, and most of the studies that do take this into account, also look at network optimization (mainly the allocation of hubs). For more information on this, we refer the reader to the literature review of research question 2 in Section 4-3. Background on the situation where prices are set in a known network is covered in 3-3.
Another related field is that of economic equilibrium studies, which was studied by Dobson and Lederer (1993) among others. Here, market competition is studied, where the main goal is getting insight into market characteristics in the long run. This requires the strategic choices of different competitors to be modeled, which is the relevant part in regard to this thesis.

1-4-2 Contributions

The first main contribution of the thesis is the proposal of a solution approach to the problem that is defined in Situation 1 of the research question, which, to the best of our knowledge, was not described in the literature before. The combination of determining prices and choosing paths for multiple commodities in a hub-and-spoke network where hubs have capacities is new in an optimization context.

The second contribution is the proposal of a new meta-heuristic algorithm for the problem that is defined in Situation 2. It is based on the principle of Local Branching (Fischetti and Lodi (2003)), and can also be used on similar problems.

Furthermore, since parts of the solution approaches that are used, are applicable in a broader context, the contribution also lies in the insight that the thesis provides into other network optimization models. It shows the impact of multiple aspects, both in the area of problem definition and input data, on the solvability of these models.

1-4-3 Challenges

What follows next is some paragraphs covering the approach followed by other researchers in dealing with the challenges that are mentioned in Section 1-3-4.

Cost modeling When considering cost modeling considerations, a point of interest is the modeling of the costs for transport between hubs. Here, because of the consolidation of flows, economies of scale have an effect. In the literature, this is generally accounted for by a discount factor for all flow between hubs independent of actual flow. However, in an influential paper, O'Kelly and Bryan (1998) say this is too simplistic and propose a non-linear function such that the travel cost per unit flow decreases as flows increase, which is then approximated by a piece-wise linear function.
Another cost aspect that has to be mentioned is the modeling of vehicle balancing costs as for example included by Meuffels (2010). Demands in different parts of the network can vary widely, which results in empty movements, for which an additional cost needs to be incorporated in the model. Meuffels (2010) resolves this by taking advantage of the fact that actual movements are modeled, and repositioning costs are set for when there are more vehicles leaving a node than there are vehicles arriving at that node.

Non-linear objective function An approach is found in the study by O'Kelly et al. (2015) and is based on the observation that the demand is uniquely determined by the cost structure and by the price-demand coupling functions. The discretized demand and price vectors are linked in a pre-processing procedure, which makes it possible to introduce new decision variables that make the objective function linear. Luer-Villagra and Marianov (2013) use a genetic algorithm because it does not require local search procedures, as the genetic operators help the algorithm to explore the solution space; solutions can be represented easily; and genetic algorithms have had good success in previous applications involving hub location problems. Eiselt and Marianov (2008) use a heuristic concentration approach because it is a procedure that has shown good results when a fixed number of facilities is to be located and is very easy to implement. This two-phase meta-heuristic uses a low complexity method in phase 1 to restrict the number of candidate locations that are, at least, local optima, and then uses an exact method or an improvement heuristic in phase 2.

Routing When incorporating path-choices for the different commodities there are two main approaches . The first is the *arc-based* approach and is used by Camargo et al. (1998) among others, and incorporates the path-finding in the main model (the model that optimizes the prices). This approach is also used in the related field of location routing problems, where instead of the price, the location of facilities is optimized. The approach uses variables on the arcs in the network, and through the use of constraints makes sure that routed goods or persons are transported from their origin to their destination within the desired service time. It is computationally expensive, since most combinations of a commodity and an arc get a decision variable and every combination of a commodity and a hub-node gets a constraint.
The other approach (Kim et al. (1999) among others) is called *path-based* and first finds all possible paths and then uses these in the main model. A natural extension of this is the use of a heuristic to define a pool of most promising candidate paths. It has a decision variable for every combination of a commodity and a path from the pool. It also has a constraint that makes sure one and only one path is chosen, this is because practical issues require that all

parcels of a commodity are transported using the same path. This principle is also used by the Ortec Infrastructure Optimizer (OIO) package, which is discussed in Section 2-3-3.
An alternative for the path-pre-generation approach is the adding of paths to the main model using the dual problem. This is used by Firat et al. (2012) on a somewhat similar problem and adds the paths as variables based on information in the optimal solutions of the LP-problem and the dual problem.

Consumer service choice Dobson and Lederer (1993) studied airline competition, and an expression is derived for calculating the demand for each path as a function of the customers' cost and prices over all paths. The customers' cost consists of two components; how much actual departure- and arrival times differ from customer's desired times, and how long the path takes. For this thesis project the first component is not applicable, and the second translates to the different service levels for parcels. The derived expression is a weighted logit function dependent on the consumer density and the characteristics of the company's (and its competitors') possible paths. The logit function is defined as $\log\left(\frac{p}{1-p}\right)$, $p \in [0,1]$, and is a popular model in the transportation literature for representing discrete choice because it provides a closed form expression and it can accommodate several attributes of the alternatives (Luer-Villagra and Marianov (2013)).

Quality of LP-relaxations The definition of an LP-relaxation and its background is covered in Section 2-1. Usually, the challenge associated with weak LP-relaxations is tackled by strengthening the bounds by adding cuts, but the size of most express shipment problems results in models that then become intractable. To enhance the process of identifying integer solutions, Armacost et al. (2004) redefined the decision variables, which results in a network design formulation whose LP-relaxation is provably stronger. They remove the flow variables as explicit decisions and embed them within the fleet assigning variables. Thereafter, they combine these variables into composite variables, which represent the selection of multiple paths that cover the demands for some subset of commodities. This results in a formulation that provides tighter bounds than the conventional one and enables very good solutions to be found quickly.

1-5 Outline of the report

In Chapter 2, the mathematical background of the methodology and the used software are described. In Chapter 3, Situation 1 of the research question, The Fixed Network, is covered. In Chapter 4, the more complex Situation 2 of the research question, The Variable Network, is covered. In Chapter 5, the findings of Chapters 3 and 4 are applied in a realistic case study. And in Chapter 6, the conclusions and possible further research are discussed.

Chapter 2

Methodology

This chapter covers an introduction to the mathematics and the software that are used in the thesis. This chapter is referred to, from various relevant parts of the report.

2-1 Mathematical programming

Mathematical programming is the field of mathematics concerned with solving *optimization problems*. These are problems on finding the maxima (or minima) of an *objective function* (see 2-1), on a set X that is defined by a set of conditions, or *constraints*, that any solution $\mathbf{x}$, which consists of a set of variables, must satisfy. These constraints can be either inequality constraints (2-2) or equality constraints (2-3).

$$\max_{\mathbf{x}=(\mathbf{w},\mathbf{z})\in(\mathbb{R}^p,\mathbb{Z}^q)} \quad f(\mathbf{x}) \tag{2-1}$$

$$s.t. \quad g_j(\mathbf{x}) \leq b_j \qquad \forall j \in J \tag{2-2}$$

$$g_k(\mathbf{x}) = b_k \qquad \forall k \in K \tag{2-3}$$

The values for the right-hand-sides of all constraints, the coefficients in $f(x)$ or $g(x)$, and the sets J and K, are called the *input data*. These can for example consist of hub capacities, flow amounts, and depots. Within the field of mathematical programming we can distinguish four main sub-fields, namely Linear Programming (LP), Non Linear Programming (NLP), Integer Programming (IP), and Non Linear Integer Programming (NLIP). In LP the variables are continuous, and the objective function and all constraints are linear, while in NLP the variables are also continuous and one or more of the constraints and/or the objective function is non-linear. In IP and NLIP the variables can only take integer values, and the problems are again respectively linear and non-linear. When a problem allows both integer and continuous variables we speak of Mixed Integer Non Linear Programming (MINLP) or Mixed Integer Linear Programming (MILP). Problems from the LP class belong to complexity class **P**. Solving these is relatively easy because the simplex or interior point method can be used,

which are very practically efficient algorithms. In general, problems from the NLP class are much harder (**NP-Hard**), but the subclass of *convex* NLP problems again belongs to **P** due to the applicability of interior point methods. A further characterization is the class of convex problems. If the solution space of a problem is convex, and the objective function is convex in a minimization problem, or concave in a maximization problem, the problem is considered convex. This generally makes an optimization problem easier to solve.

The problems that are solved in this thesis are mostly in the class of MILP problems. For these, polynomial time algorithms are in general not possible (unless **P**=**NP**), and they are mostly solved by LP-based Branch And Bound (B&B) algorithms.

B&B-algorithm In this algorithm, a very important concept is that of the *LP-relaxation*. This is the model that arises from a model with at least one integer variable, if all constraints that require integrality are relaxed. All variables are then continuous, but the minimum and maximum values of the variables are the same as in the original model.
In the basic version of this algorithm, the set of feasible solutions is explored systematically such that (likely) not all feasible solutions have to be visited to find the guaranteed best one. It uses a tree-like approach with a slightly modified model at each node, where the root-node contains the LP-relaxation $LP^{(0)}$ of the original model P. This explanation is based on a maximization problem. At every node k in the tree, the algorithm starts by finding the optimal solution $\mathbf{x}_{(k)}$ to the LP-relaxation $LP^{(k)}$ of the model at node k. Now there are four possible outcomes:

1. All variables that are constrained to be integer in P, are also integer in $\mathbf{x}_{(k)}$.
   $\rightarrow$ $x_{(k)}$ is a feasible solution to P, and the objective value $f(\mathbf{x}_{(k)})$ is a lower bound on the optimal solution value.
2. $LP^{(k)}$ is infeasible.
   $\rightarrow$ The tree is *pruned* (discarded) at node k, as no feasible solution to P can exists in this part of the tree.
3. The objective value $f(\mathbf{x}_{(k)})$ is smaller than the largest lower bound.
   $\rightarrow$ The tree is pruned at node k, as no better solution can exists lower in the tree.
4. A variable that was constrained to be integer in P, has a fractional value in $\mathbf{x}_{(k)}$, and we are not in case 3.
   $\rightarrow$ The feasible region is split into two parts (branches in the tree), and this excludes the fractional solution in any node in the tree below k.

Next, the node is selected that will be processed after node k. For this, several selection strategies are possible such as Most Infeasible Branching (this branching strategy chooses the variable whose fractional part is closest to 0.5).
Furthermore, along the way heuristic procedures are used to find solutions that satisfy the original model including all integrality constraints. An example of such a heuristic is a rounding procedure that smartly chooses an integral value for every fractional variable. These solutions also become lower bounds on the optimal solution to P. The solution having the best lower bound is called the incumbent solution. The upper bound is obtained by taking the

minimum of the optimal objective values of all current leaf nodes in the tree. The difference between the largest lower bound and smallest upper bound is called the *LP-gap*, and this plays an important role in deciding if and when the algorithm is stopped. It is a guarantee on the quality of the incumbent solution, and if it reaches zero, the incumbent solution is guaranteed to be optimal.

Branch And Cut (B&C)-algorithm An extension of the B&B-algorithm is the B&C-algorithm. In this algorithm, *cutting planes* (or cuts) are added that *strengthen* the LP-relaxation. This means that a constraint is added to the model that cuts off a part of the feasible region of the LP-relaxation, without cutting off any integer points. This is schematically shown in Figure 2-1. This is done in the original formulation (the root node), as well as in the nodes that are lower in the tree. A consequence of this is that the upper bounds get better, which means more branches can be pruned, which in turn reduces the time needed for the entire algorithm. Although adding cuts can improve the performance of the algorithm, adding too many of them will result in "heavy" models for which the increase in node processing time is such that it defeats the advantage of increased upper bounds.

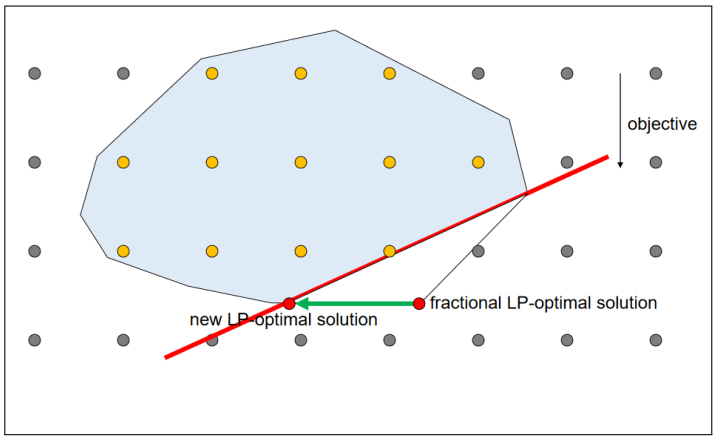


Figure 2-1: Schematic representation of a cutting plane (GUROBI Optimization (2018)).

State-of-the-art algorithms GUROBI 7.0 (Gurobi Optimization (2018)) and CPLEX 12.6.3 (IBM Corporation (2018)) are the modern general-purpose MILP solvers that are used in this thesis. They both use a similar method based on the B&C algorithm. This is shown in Figure 2-2.

The key features of the method are the following:

1. Pre-solving: The solver tries to detect special characteristics and implement certain changes in the input, that will most likely lead to a better performance of the solution process. This is most extensively, but not exclusively, done in the root-node. Examples are the removal of redundant constraints, identifying disconnected components (in graph models), and exploiting integrality. The pre-solve procedures are heuristic in nature,

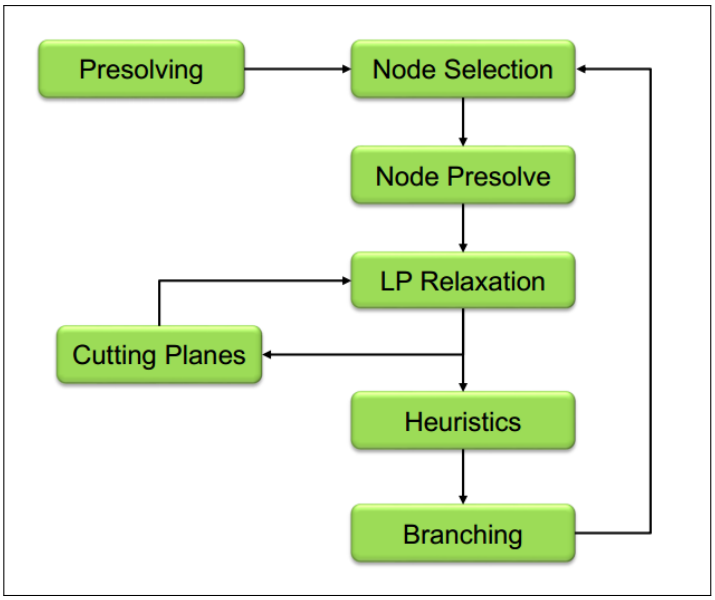


Figure 2-2: Overview of the algorithm that is used by GUROBI (GUROBI Optimization (2018))

i.e., they do not exhaustively search for all possible reductions.

2. Cutting plane generation: The discovery and implementation of inequalities that refine the existing problem formulation and reduce the feasible set of solutions. The cuts that are added during the solution process, that is described in Chapter 3, are shown below. For more details we refer the reader to Marchand et al. (2001) or Nemhauser and Wolsey (1988).
    - Implied bound.
    - Gomory Mixed Integer Cut.
    - Cover Cut.
    - Mixed Integer Rounding (MIR) Cut.
    - Flow Cover Cut.
3. Branching strategies: Sophisticated ways to partition the problem into two smaller problems, based on the removal of a fractional solution of the LP-relaxation.
4. Node selection strategies: Sophisticated ways to select the next node to process. The goal in this step is to move to the "best" part of the tree as quickly as possible, without knowing up front what the lower branches will look like.
5. Primal heuristics: The exploitation of problem characteristics to find good quality solutions early in the tree, for example by rounding heuristics, or by using local search algorithms.

Performance variability In this thesis, a lot of time and effort is spent on determining the performance of solving MILP models. By this, we mainly mean the time it takes until an optimal (or good enough) solution is found. However, performance variability is a known and serious problem. Danna (2008) defines this as "a change in performance (solving times,

number of nodes, or number of iterations) for the same model created by a change in the solver or the environment that is seemingly performance neutral". While there is reasonable understanding of the factors contributing to performance variability in general, the specific causes are not well known. It is still difficult to know whether a model is likely to be affected, and to what degree, by performance variability issues, except by trial and error.

2-2 Meta-heuristics

Meta-heuristics (e.g. Gendreau and Potvin (2010)) is the name for a very diverse collection of algorithms that guide the search for a solution in a broad range of optimization problems. The goal is to efficiently explore the solution space to find good quality solutions. They primarily consist of an interaction between local improvement procedures, which are mostly based on the establishment of neighborhoods around the current solution, and higher level strategies, that make up a process that can escape from local optima and tries to find a global optimum of a solution space. An important characteristic is that the algorithms, in general, cannot provide a guarantee on the quality of the found solution.

Variable Neighborhood Search (VNS) In VNS, neighborhood structures are explored in search of a new solution, which is only accepted as the new solution if it is better than the current one. These neighborhoods are variable, and they change (they usually get bigger) with every iteration of the local search. When a certain threshold is reached, the current solution is accepted as the local optimum, and the perturbation phase is started to get out of the local "valley".

TABU search This is a meta-heuristic search method that uses memory structures for already visited solutions. This is needed because it allows solutions that are worse than the current solution to be accepted to get out of local optima. Without the memory function there is a risk of getting stuck in a cycle of already visited solutions.

Large Neighborhood Search (LNS) In general, one can say that when performing local search, using large neighborhoods leads to good quality solutions, but is very time-consuming. In LNS, an initial solution is iteratively improved by smartly destroying and then repairing a part of the current solution. The neighborhood is then defined as the set of all solutions that can be constructed from the partly destroyed solution. This neighborhood is very large in most cases, and repairing the solution can be viewed as searching the neighborhood heuristically. This way, large neighborhoods can be searched, but within a relatively short amount of time.

2-3 Software

2-3-1 AIMMS

Advanced Interactive Multidimensional Modeling System (AIMMS) is a commercial modeling and optimization software environment built for optimization and scheduling problems. It is mainly used for using mathematical optimization techniques to improve operational, tactical and strategical decisions in several fields. It was designed to have a simple and visual interface, and it is linked to several solvers such as CPLEX, Gurobi, and MOSEK.

2-3-2 MILP solvers

MILP solvers are the most used software in this thesis. Although they were originally intended to be black-box exact tools, they can serve other purposes as well. They can for instance serve as a framework for algorithms combining both exact and heuristic elements, be used just to find a feasible solution, or approach the optimum until a specified bound is reached. They have a multitude of options which can be exploited in the context of this thesis. Their basic approach is the B&C algorithm.

CPLEX This is a widely used commercial optimization package by IBM that can also be used as a stand-alone package. It has interfaces for C++, C#, and Java languages, and is able to solve very large instances of problems in the classes LP, IP, MILP, Quadratic Programming (QP) and Quadratically Constrained Programming (QCP). For solving problems in the MILP class, which is most relevant for this thesis, it uses the B&B-algorithm in combination with a multitude of techniques such as primal or dual variants of the simplex method or the barrier interior point method, and the strengthening of the problem by generating several kinds of cutting planes.

Gurobi This thesis uses GUROBI as the solver for the models after they are implemented in AIMMS. Its basic methods are very similar to those of CPLEX, but it performs better than CPLEX (or any other MILP solver) in almost every aspect, as shown by Mittelmann (2018). Unfortunately, the reasons behind this are classified, and nothing can be said that is not pure speculation.

2-3-3 Ortec Infrastructure Optimizer (OIO)

OIO is the tool used by Ortec to optimize the network of express service providers. It has four parts focusing on the depot and area part, the hub and line-haul part, the line-haul scheduling part, and the Pick Up and Delivery (PUD) tour scheduling. For this thesis the tool was used as a source of inspiration in dealing with modeling and implementation challenges, as a reference method to check if the results of the models give reasonable overall results, and as a path generator.

Path generation For the generation of paths, a multitude of options is available such as setting the maximum number of hubs that can be visited on a path, selecting the arcs that are allowed to be used, and requiring direct paths for loads above a certain threshold. It then algorithmically calculates all possible paths for a commodity that meets the network defined time constraints and the service requirements. The best routes, based on a specified maximum number of routes per commodity, are exported. The criteria for determining the best routes are handling and transport costs.

Chapter 3

The Fixed Network

3-1 Introduction to the fixed network

This chapter covers Situation 1 of the research question, which is where most parts of the network are fixed. This means that most structural parts of the network are known, and these cannot be changed. But some other parts, for which changes have a less drastic effect on the rest of the network, and which are easier to implement in practice, *can* be changed. More particularly, the number of hubs and their locations are fixed, and they have a fixed capacity, depots have a fixed location and cannot be closed, and arcs are allowed to be opened or closed. An overview of this is shown in Table 3-1.

Table 3-1: Implementation of variables in the fixed network.

Variable	Range	Domain	Comment
Market share	Continuous and bounded	Set of potential paths	Also acts as path-choice variable
Arcs	Binary	Set of potential arcs	-
Depots	-	Set of fixed depots	
Hubs	-	Set of fixed hubs	Capacitated

For a visual representation of all possibilities and consequences, we refer the reader to Appendix A-1. The main difficulty in the model is the interplay between investment and operational costs, the multi-commodity aspect, and the presence of capacity constraints.

The rest of the chapter is arranged as follows. In Section 3-3, a review of related literature is covered, along with the contributions of this chapter and some challenges as presented in the previous chapter. In section 3-2-1, a mathematical formulation of the problem is given. This is followed by the solution method in Section 3-4, and the sensitivity analysis in Section 3-5.

3-2 Mathematical model

In this section the formulation of the mathematical model is given, followed by an explanation of the modeling choices.

3-2-1 Non-linear problem formulation

Sets

K	Set of commodities,
P	set of potential paths (every path can only be used by one commodity),
$P^{(k)}$	partition of P containing all potential paths that can be used by commodity k
S	set of potential arcs,
$P^{(s)}$	subset of P containing all potential paths that use arc s
H	set of hubs,
$P^{(h)}$	subset of P containing all potential paths that use hub h
D	set of depots,
$P^{O(d)}$	subset of P containing all potential paths that have depot d as its origin,
$P^{D(d)}$	subset of P containing all potential paths that have depot d as its destination.

Parameters

$x_{\min}(k)$	Minimum price of commodity k ,
$x_{\max}(k)$	maximum price of commodity k ,
$m_k(x_k)$	market share (range: $[0, a]$, $a \in (0, 1]$ limits the range to where the relation is valid) of commodity k as a function of its price x_k,
f_k^w	total amount of flow of commodity k when it has 100% market share, expressed in units of weight (w),
f_k^{pcs}	total amount of flow of commodity k when it has 100% market share, expressed in number of pieces (pcs),
c_p^w	variable cost of path p per unit of weight (w): sum of the variable costs of all arcs that are used by that path,
c_p^{pcs}	variable cost of path p per piece(pcs): sum of the variable costs of all hubs that are visited on the path,
$\widehat{c}_s$	cost of opening arc s,
$\breve{c}_{(d_1,d_2)}$	variable balancing cost per unit of weight between depots d_1 and d_2,
C_h	capacity of hub h expressed in number of pieces.

Decision variables

x_k Price of commodity k,

y_{kp} $\begin{cases} 1 \text{ if commodity } k \text{ uses path } p \\ 0 \text{ otherwise} \end{cases}$,

$\breve{y}_{(d_1,d_2)}$ quantity expressed in units of weight for depot-depot pair (d_1, d_2) that represents the amount of balancing of flow,

$\widehat{z}_s$ $\begin{cases} 1 \text{ if arc } s \text{ is open} \\ 0 \text{ otherwise} \end{cases}$.

Model formulation

$$\max_{x,y,\breve{y},\widehat{z}} \quad \sum_{k\in K}\sum_{p\in P} m_k(x_k)\left({f_k}^w\,(x_k - y_{kp}{c_p}^w) - {f_k}^{pcs} y_{kp}{c_p}^{pcs}\right) - \sum_{(d_1,d_2)\in(D\times D)} \breve{c}_{(d_1,d_2)}\breve{y}_{(d_1,d_2)} - \sum_{s\in S}\widehat{c}_s\widehat{z}_s \tag{3-1}$$

$s.t.$

$$x_{\min}(k) \le x_k \le x_{\max}(k) \qquad \forall k \in K \tag{3-2}$$

$$y_{kp} \in \{0,1\} \qquad \forall k \in K, p \in P \tag{3-3}$$

$$\sum_{p\in P} y_{kp} = 1 \qquad \forall k \in K \tag{3-4}$$

$$\widehat{z}_s \in \{0,1\} \qquad \forall s \in S \tag{3-5}$$

$$m(x_k)\cdot y_{kp} = 0 \quad \text{if} z_s = 0 \qquad \forall s \in S, \forall p \in P^{(s)}, k \text{ s.t. } p \in P^k \tag{3-6}$$

$$\sum_{k\in K}\sum_{p\in P^{(k)}\cap P^{(h)}} f_k^{pcs}\cdot m(x_k)\cdot y_{kp} \le C_h \qquad \forall h \in H \tag{3-7}$$

$$\breve{y}_{(d_1,d_2)} \geqslant 0 \qquad \forall (d_1,d_2) \in (D\times D) \tag{3-8}$$

$$\sum_{k\in K^{O(d)}} f_k^w m(x_k) - \sum_{k\in K^{D(d)}} f_k^w m(x_k) = \sum_{d_1\in D}\breve{y}_{(d,d_1)} - \sum_{d_1\in D}\breve{y}_{(d_1,d)} \qquad \forall d \in D \tag{3-9}$$

The objective is to maximize the total profit, which is the sum of all revenues minus the sum of all costs. The revenue of a commodity k is represented by

$$m_k(x_k){f_k}^w x_k.$$

The variables costs of path p, as a result of the flow of commodity k, are represented by

$$m_k(x_k){f_k}^w y_{kp}{c_p}^w \quad \text{and} \quad m_k(x_k){f_k}^{pcs} y_{kp}{c_p}^{pcs}.$$

The costs of the repositioning and the arcs are represented by

$$\sum_{(d_1,d_2)\in(D\times D)} \breve{c}_{(d_1,d_2)}\breve{y}_{(d_1,d_2)} \quad \text{and} \quad \sum_{s\in S}\widehat{c}_s\widehat{z}_s.$$

Constraints (3-2) through (3-5) enforce the range of all variables. Constraints (3-4) ensure that at most one path is chosen per commodity. Constraints (3-6) ensure that no flow is possible over an arc when it is closed. Constraints (3-7) are the capacity constraints at the hubs. Constraints (3-9) ensure an imbalance in flow between depots is penalized.
A useful visualization to understand the model, is that for every commodity a price and a path is chosen, and these in turn determine the decisions on arcs. And all of this is influenced by the capacity and balancing constraints. Of course, when the model is solved, these decisions are made simultaneously. More background on the problem is found in Section 3-2-3.

3-2-2 Modeling choices

Here, the choices are covered that were made in translating the reality of an express network into the mathematical model as given in the previous subsection. Aspects that are excluded from the model are covered in the paragraph on the scope of the thesis in Section 1-3-4. The itemization that follows now, treats aspects that apply to both situations as defined in the research question.

- The path-based formulation, opposed to the arc-based formulation, as described in Section 1-4-3 is used for modeling the routing of parcels. The main reason for this is that, compared to the arc-based formulation, it is likely less computationally expensive because of the better LP-gap (as shown in Kuiteing et al. (2015)), and it provides more flexibility for the generation and admission of paths since the path generation is done separately from the main model. For the generation of paths, OIO is used, of which a description is given in Subsection 2-3-3. Afterwards a selection is made that determines the set of potential paths. This selection procedure is covered later in the chapter.
- Arc costs are assumed to have a fixed and a variable part (as in Campbell (1994)). This represents a flow threshold, modeling the fact that the "first" parcel that uses an arc requires one unit of transportation (e.g. a truck) to be used. Furthermore, the variable part is linear with respect to distance and to the amount of flow (given in terms of weight). This is an approximation of actual costs which has a "step-function-like" character because of the fact that discrete units of transportation will be used. Since in reality, there is no real limit to the number of trucks that can be used on an arc, the arcs in the model are uncapacitated.
- For the depot and hub costs a similar principle is applied. Here the fixed cost represents the cost of opening a depot, and the variable costs are linear with respect to the amount of flow in terms of pieces. This last part not only accounts for handling costs at the depot, but also has an added part, which is an approximation of the costs of the PUD-process as defined in Section 1-2.
- Depots are assumed to have unlimited throughput capacity (i.e. they are uncapacitated).
- Prices are set per commodity.
- As discussed before, a requirement on the price-market share relation is that at the maximum price, the market share of a commodity is zero. If a commodity is not profitable and its flow should not be transported in the optimal solution, setting the

price to its maximum will achieve this.
- Customer choice is simply modeled by only using the price-market share relation for an commodity. No interdependence between combinations is used.
- Vehicle balancing is approximated (i.e. actual truck movements are not modeled) by transporting an amount of flow between depots such that the amount of flow that comes in, is equal to the amount of flow that goes out for every depot. For this, extra direct paths are used, and the costs of this re-balancing are the same as the variable costs of a direct arc.

3-2-3 Background of the problem

The problem formulation as presented in the previous subsection is in the class of MINLP problems. In this case, we notice that the problem has a non-linear objective function, and some constraints are also non-linear. It is well-known that, in general, these problems are NP-hard.

The non-linear terms in the objective function are composed of the product of the price and the market share, which is dependent on the price. Furthermore, there are multi-linear terms, which are the product of the path-choice variable and the market share.
To determine the best approach for solving the problem, it is important whether or not the objective function, after dropping all integrality constraints, is concave. Both cases allow different solution methods, which will likely have very different solution times. A main reason for this is that for concave functions, any local maximum is also a global maximum. In our case, this will depend on the relation between the price and the market share. As described in Subsection 1-3-2, we assume that the relation between the price and the market share is a non-increasing and continuous function. From this, we get the following result.

Proposition 3-2.1. *The objective function of the original non-linear formulation (3-1) is not concave on the range defined by the constraints, when the relation $m(x)$ between price and market share is **any** polynomial.*

Proof. Take a multi-linear term $f(x)$ of the objective function that is the product of a non-negative constant $\tilde{c}$, the path-choice variable y with range $[0, 1]$, and the non-negative market share $m(x)$. The latter is a polynomial with degree n describing the relation between the price x and market share which can be represented as $m(x) = \sum_{i=0}^{n} c_i x^i$. Then $f(x) = \tilde{c}y \cdot \sum_{i=0}^{n} c_i x^i$ and the Hessian matrix of this function is

$$\mathbf{H}(f) = \begin{bmatrix} f_{xx} & f_{xy} \\ f_{yx} & f_{yy} \end{bmatrix} = \begin{bmatrix} \sum_{i=2}^{n} c_i \cdot i \cdot (i-1) \cdot x^{i-2} & \sum_{i=1}^{n} c_i \cdot i \cdot x^{i-1} \\ \sum_{i=1}^{n} c_i \cdot i \cdot x^{i-1} & 0 \end{bmatrix}$$

The function $f(x)$ is concave if and only if, for all $z_1, z_2 \in \mathbb{R}$, $h(\mathbf{z}) = \begin{bmatrix} z_1 & z_2 \end{bmatrix} \mathbf{H} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \leqslant 0$.

Now if we let $z_2 > -\frac{z_1}{2} \cdot \frac{f_{xx}}{f_{xy}}$ we get that

$$\begin{aligned} h(\mathbf{z}) =& {z_1}^2 \cdot f_{xx} + 2 z_1 z_2 f_{xy} \\ &> {z_1}^2 \cdot f_{xx} + 2 z_1 f_{xy} \cdot -\frac{z_1}{2} \cdot \frac{f_{xx}}{f_{xy}} \\ &= {z_1}^2 \cdot f_{xx} - {z_1}^2 f_{xx} \\ &= 0 \end{aligned}$$

And so the proposition follows. □

Another feature of the problem are the "on/off" constraints (3-6), which impose whether there can be flow on the arcs or not. They use the binary variables for the arcs, which are the only non-continuous variables in the model. Implementing this type of constraint in a modeling language like AIMMS requires some attention because it can be done in several ways that have different computational effects. Some possibilities for this are discussed in Subsection 3-4-6.

Something that is mathematically interesting is that finding a feasible solution to the problem is always possible. No matter what the input data is, by which we mean all sets and parameters as they are defined in the previous section, setting all market shares to zero is a feasible solution. This is different from conventional problems in this field, as in these problems, it is typically not an option to *not* transport a flow, when this is preferred due to the constraints. This is especially relevant when trying to strengthen the model formulation. More on this is covered in Subsection 3-4-6.

The balancing constraints have two possible consequences that can occur simultaneously. Firstly, a variable $\breve{y}_{(d_1,d_2)}$ gets a positive value, which decreases the objective value. This approximates the repositioning cost of vehicles. Secondly, the flows between depots are adjusted through the market shares of the commodities, such that no repositioning is needed.

3-3 Literature

This section expands upon the general literature section in Chapter 1 and concerns the first research question. It covers its relation to other research and potential solutions to some challenges as presented in Section 1-4.

3-3-1 Context

A field of study that is closely related to the current research question is airline scheduling and routing. Dobson and Lederer (1993) studied the competitive choice of prices and flight schedules for the transshipment of people instead of parcels over a HS-system. The main differences compared to this thesis are a somewhat different cost structure (e.g. no variable costs for the units to be transported, in this case passengers), and the fact that customers

have the choice to take different paths from their origin- to their destination-node. It is worth noting that in the study by Dobson and Lederer, profit optimization is not the main objective. It is only applied to the different companies in the model, and the goal is getting a better understanding of airline competition and its market characteristics.
Another related field is that of the NPP, which was introduced by Labbe et al. (1998). It determines revenue-maximizing tolls on a transportation network, and it takes into account that the users are assigned to the cheapest path that they can take. The main difference with this thesis' problem description is that prices are set for individual arcs, instead of origin-destination pairs. Kuiteing et al. (2015) extended this problem to a situation where demand is elastic and linearly decreases as the transportation cost increases. They focus mainly on the structure and properties of the problem, including its theoretical complexity, and conclude that even in a simple case, where prices are dependent on the length of the corresponding arc and a single pricing decision over the entire network, the problem is NP-hard. Also, a computational sensitivity analysis for several key parameters is done.

3-3-2 Challenges

In this subsection we cover the approach followed by other researchers in dealing with the challenges (as they were defined in Section 1-4-3) that are specific to this chapter. Therefore, from the challenges, we only cover the choice of the solution method here.

Solution methods The study that is most closely related to this chapter was done by Dobson and Lederer (1993), of which a short problem description is covered above in Subsection 3-3-1. In this study, the optimal prices were found by writing out the Lagrangian dual (a different form of the problem found by dualizing some of the constraints) and the Kuhn-Tucker conditions, which are necessary conditions for a solution in nonlinear optimization to be optimal. Afterwards a sub-gradient optimization procedure, which is an iterative method that works well on large scale problems, is used to find the optimal prices.

Another approach is used by Brotcorne et al. (2011). This study focuses on an improved exact algorithm for addressing an NP-hard network pricing problem. The method involves an efficient and partial generation of candidate solutions, a recursive scheme for generating improved upper bounds, and a column generation procedure for solving the network-structured sub problems.

3-4 Solution approach

In this section we cover the approach in finding a good solution to the model that was formulated in the last section.

3-4-1 General approach possibilities

The terms in the objective function of the original non-linear formulation that are a multiplication of the market share and the price, are of degree two or more, because in reality the price-market share relation will not be a constant function. These terms increase the complexity of the problem a lot, making it unlikely that finding the optimal solution is possible in a reasonable amount of time. Furthermore, practical experience at ORTEC has taught that finding the optimal solution of similar *linear* problems is also very unlikely for networks of realistic size. This is why an approach will need to be found that approximates the optimal solution within acceptable solution time.
In the case of this thesis there are several main approaches that can be used, or combined, to get the best results. The first is modifying the problem such that algorithms can be used that, in theory, can find the optimal solution to this modified problem. The solution that is then found is an approximation of the optimal solution to the original problem.
The second approach is using some sort of (meta-)heuristic method. This is an approach based on either the practical meaning or the mathematical structure of the problem that (hopefully) works well in practice, but cannot guarantee the quality of the solution it finds.
The third approach is using approximation algorithms, for which a provable guarantee exists on the relation between the found solution and the optimal solution.
The possibilities can be subdivided into three main categories that are based on the manner in which the method deals with the non-linearity of the objective function. The first possibility is finding an approach to solve the unmodified MINLP problem. The last two possibilities use different methods to transform the problem into a MILP problem. In the paragraphs that follow, all three approaches are covered.

MINLP methods In this approach, the problem formulation is not altered, and methods are used that are able to solve or approximate NLP problems (convex *and* non-convex). These methods are, in practice, only applicable to small problems, since they are intended to work on the full range of NLP problems and this makes them relatively slow for most large problems. An advantage of the problem formulation is that, as we will see later in this section, it can be reformulated into a convex problem. This makes it significantly easier to solve than general non-convex problems. However, also algorithms intended for solving large scale instances of convex MINLP problems in a practical amount of time, are very likely still out of reach. But in specific instances, even more specialized methods can be used; if the price-market share relation is a linear function, the problem is in the class of convex MIQP problems. In this case, a promising approach is the QP based branch-and-bound method. Here, the relaxations at the nodes of the branch-and-bound tree are in the class of convex QP problems, which are harder to solve than LP problems, but are still relatively easy.

Linearization Another possible method is transforming the model into a MILP model. This modified model can be solved by using methods that are developed for this specific purpose. A big advantage of this approach is that the field of solving MILP problems is a very well

researched and mature field, and there are a lot of known methods that are fast, reliable, and able to solve very large problems.
There are two terms in the objective function that prevent the model from being in the MILP class. The first is the multi-linear term, which is the product of the path-choice variable and the market share. This can be easily transformed to a linear term, and this is covered in the next subsection. The second term that prevents the model from being in the MILP class is the product of the price and the market share. These non-linear terms can be approximated, and this results in an optimal solution that is different from the optimal solution to the non-linear problem.
In general, approximating a non-linear function is done by generating a collection of simplices that approximates the function values in every point, such that the error is minimized. An advantage here is that the non-linear terms are univariate. The number of required simplices increases sharply with an increase in dimensions (Lin et al. (2013a)), so a relatively small amount of them is needed in our one-dimensional case. Furthermore, the complete objective function does not need to be approximated at once, but all terms can be approximated separately by a piece-wise linear function.
For linearizing a non-linear function many methods are possible. All of the methods that are applicable to general univariate functions introduce extra binary variables (Lin et al. (2013a)). From the perspective of solving the model, this is very disadvantageous since these variables make the MILP model more complex and harder to solve. This is because integer variables often slow down a MILP solver. They tend to make the branch-and-bound tree deep and unbalanced because branching is more complicated. Also, they do not improve the bound of the LP-relaxation (Lin et al. (2013b)). However, because the objective function is concave, we can use a method that does not involve extra binary variables. The concavity is a result of the assumptions on the relation between the price and market share as they are given in 1-3-2. This is shown in Proposition 3-4.1.

Proposition 3-4.1. *Let x be the price, and $g(x)$ the relation between the price and the market share. Then $g(x) = x \cdot f(x)$ is a concave polynomial on the range where $g(x)$ is defined.*

Proof. For the domain in which $g(x)$ is defined, we know it is a concave and non-increasing polynomial. Assume it is of the form $\sum_{i=0}^{n} c_i \cdot x^i$. Then we have that

$$f(x) = x \cdot g(x) = x \cdot \sum_{i=0}^{n} c_i \cdot x^i = \sum_{i=0}^{n} c_i \cdot x^{i+1}$$

The second derivative of this function can be written as

$$f''(x) = \sum_{i=1}^{n} c_i \cdot (i+1) \cdot i \cdot x^{i-1} = x \sum_{i=1}^{n} c_i \cdot (i+1) \cdot i \cdot x^{i-2}$$

If this function is non-positive, we know $f(x)$ is concave. Because $g(x)$ is concave we can say that the summation from the function above is non-positive:

$$\sum_{i=1}^{n} c_i \cdot (i+1) \cdot i \cdot x^{i-2} \leq \sum_{i=1}^{n} c_i \cdot (i-1) \cdot i \cdot x^{i-2} = g''(x) \leq 0$$

And because x is positive we get

$$f''(x) = x \sum_{i=1}^{n} c_i \cdot (i+1) \cdot i \cdot x^{i-2} \leq 0,$$

which proves the proposition. □

Because all other terms in the objective function are linear, and a sum over linear and concave functions is again concave, we can say the objective function is concave.
This makes it possible to approximate the function by linear functions as shown in Figure 3-1. The way in which the approximation can be integrated in the model is covered later in this section. This approach will have a significantly less negative effect on the solution time than the other linearization approaches that were discussed.

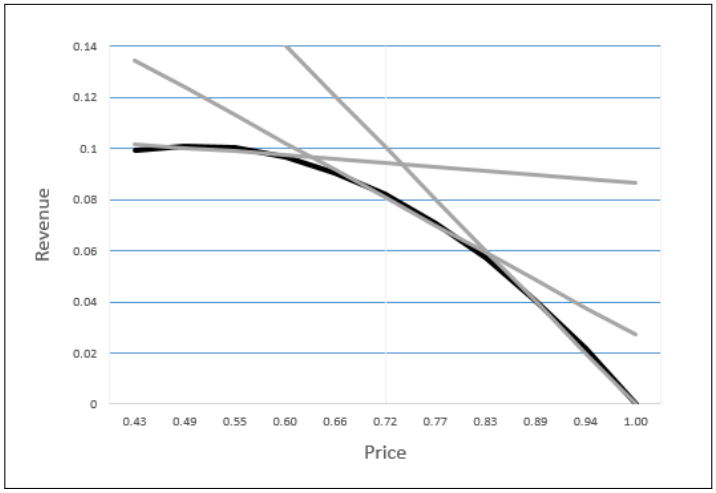


Figure 3-1: A linearization of the revenue function using 3 lines.

Discretization Another possibility to transform the non-linear model into a MILP model is using a set of discrete points, instead of a continuous range, as the domain of the non-linear function. A binary variable is then assigned to every point in the set, and constraints are added that ensure one and only one point is chosen. The resulting model is a so-called pseudo-boolean optimization model, which means all variables are binary, and the objective function is allowed to be non-binary.
The model would then be formulated in the following way. For every path, a set of fixed market shares is chosen (for instance with a regular interval from the minimum to the maximum). Every combination of a commodity and a market share is assigned a binary variable, and constraints are added that ensure one and only one path and market share are chosen per commodity. For every combination of a path and a market share, we can now compute the revenues from the non-linear function, and all variable costs. These are combined into a set of profits, which are used as the coefficients of the binary variables in the objective function. As a result of this, both the non-linear terms (product of the price and market share), and the multi-linear terms (product of the market share and the path-choice) are now linear.

3-4-2 A MILP reformulation

What follows now is an explanation of how the MINLP model is approximated by a MILP model.

General approach From the options that were presented in the previous subsection, we have chosen to use the linearization approach, and as mentioned there, the main consequence of this is that the resulting model is a MILP model. The main reasons for this decision are:

1. The use of NLP methods is not practically possible since some tests we have performed on relatively simple models revealed that the solution process of these methods require too much time. This is something that is also confirmed by experts in the field. Furthermore, a benefit of not choosing the NLP approach is the fact that for the other two approaches, linearization and discretization, the solution time can be influenced by the coarsity of the approximation/grid.
2. The linearized approach performs better than the discretized approach on basic versions of the model.
3. No additional binary variables are introduced in the linearization approach, which means the arc variables are the only integer variables. The path-choice variables are also integer, but they can be relaxed. This will be explained later in this subsection. This means that when the arc variables are fixed, the model is a very quickly solvable LP-model. This opens the door to decomposition and local search techniques.

Changing the main variables As the main set of variables in the model, we use the market share $\dot{x}$ instead of the price x. These two are directly related through the price-market share relation. If we do not make this transformation (i.e. the price is used as the main variable), and the price-market share relation is non-linear, Constraints 3-6 are non-linear. This is prevented by using the market share as the main variable. Now, the only non-linear elements in the model are the non-linear terms in the objective function, which can be linearized by the procedure that is explained below. Because we no longer use the market share as a function of the price, but vice versa, we introduce the price function $P(\dot{x}) = m^{-1}(x)$, which is the inverse of the market share function.

Linearization We require the price-market share relation to have certain characteristics that have already been mentioned in Subsection 1-3-2 and are further explained in Section 3-4-1. A consequence of these characteristics is that the linearization approach can be applied, without limiting the problem in any significant way. In the rest of the report we will, for the sake of simplicity and without loss of generality, assume that the price-market share relation is linear. As a result of this, the non-linear terms in the objective function are quadratic.

The linearization of the objective function is implemented in the following way: We start with

a quadratic term in the objective function that represents the revenue of a commodity k:

$$R_k(\dot{x}_k) = \dot{x}_k \cdot {f_k}^w \cdot P_k(\dot{x}_k),$$

where $\dot{x}_k = \sum_{p \in P^{(k)}} \dot{x}_p$, i.e. the sum of the market shares of all paths that a commodity can use. This function was rewritten in comparison to Equation 3-1 because the main variable was changed from the price x to the market share $\dot{x}$. The function is then approximated by a set of linear functions (as explained in Section 3-4-1):

$$r_{k,l}(\dot{x}_k) = a_{k,l}\dot{x}_k + b_{k,l} \qquad l \in \mathbb{N}, 1 \leq l \leq n,$$

where n is the number of linear functions that are used to approximate $R_k(\dot{x}_k)$, $a_{k,l}$ is called the *gradient* of linear function l, and $b_{k,l}$ is called the *offset*. Now, new variables y_k are introduced that replace the quadratic terms in the objective function. Constraints

$$r_{k,l}(y_k) \leq a_{k,l}y_k + b_{k,l} \qquad \forall k \in K, l \in L$$

are added to the model. $R_k(\dot{x}_k)$ has positive coefficients in the objective function, so the new variables y_k that replace them also have positive coefficients, and thus in any optimal solution they are maximized so that the linearization constraints are tight for every y_k. The constraints are dependent on the chosen market shares $\dot{x}_k$. So in every optimal solution, these y_k now take on the value of the approximation of the quadratic function.

The multi-linear terms To deal with the product $\dot{x}_k \cdot y_{kp}$ of the market share variable and the binary path-choice variable, which is present in both the objective function and Constraints 3-6 and 3-7, we construct new composite variables $\dot{x}_p$ that take on the value of the product. The index of commodity k in y_{kp} is no longer necessary if every path can only be used by one commodity. So instead of variables that determine the market share per commodity, we now have variables that determine the market share of every path. These are accompanied by constraints that ensure the sum of the market shares over all paths p that can be used by a commodity k, is less than or equal to the maximum market share:

$$\sum_{p \in P^{(k)}} \dot{x}_p \leq \dot{x}_{\max}(k)$$

A consequence of using the market share as the main variable and the introduction of these composite variables $\dot{x}_p$, is that choosing zero market share for a path now means that the variable is also zero (i.e. $\dot{x} = 0$). This is not the case when the variable is the price, where zero market share is achieved through setting the price to the maximum $m(x_{max}) = 0$. This is an advantage when the model is programmed because the options of not choosing a path, and choosing zero market share on a path, is both done by setting the variable $\dot{x}$ to zero. Where in the other case these two options have to be programmed into the same result, namely no market share on the path.

Path relaxation The constraints that ensure that one and only one path is chosen per commodity, can be relaxed into constraints that ensure that the total market share over the potential paths for a commodity is at most the maximum market share. This makes the model easier to solve, since the constraints and binary variables needed for enforcing the path-choice considerably increase the complexity. In the case when the hubs have unlimited capacity, this does not make a difference in the optimal solution, since for a commodity one path will be the cheapest, and hence the maximal amount of flow will be sent over this path (the arcs also have unlimited capacity). This only changes in the case that the capacity of a hub along a cheapest path is reached. In this case, the flow might be split over multiple paths and the optimal solution is different. In Proposition 3-4.3, we prove a characteristic of this difference. For this, the following observation is insightful.

Observation 3-4.2. We will show by an example that, in a solution of problem formulation (3-12)-(3-20) where the path choice is relaxed, and in which there are flows that are split over more than one path, there is a way to rearrange the flows such that we get a new feasible solution, that has the same total revenue and a different total cost. To get a feasible solution we require that the total transported flow over every hub is the same as in the original solution. And to have the same revenues, the total transported flow for every commodity should stay the same.
A schematic representation of a network is shown in Figure 3-2. Assume we have commodities a, b, c, ..., and their flow is split over paths $p_{(a,1)}$, $p_{(a,2)}$, $p_{(b,1)}$, ... Also assume that $p_{(a,1)}$ and $p_{(b,1)}$ both use the same hub h_1. If we create a new solution in which we increase the flow on $p_{(a,1)}$ by a certain amount, and decrease the flow on $p_{(b,1)}$ by the same amount, or vice versa, the amount of flow in hub h_1 stays the same. As a result of this, we can be sure the capacity constraint will not be violated, and thus that the new solution is feasible. As mentioned before, we aim to keep the revenues the same, so the sum of the flows over all paths that commodity a uses, should stay the same. That means that if the flow increases on path $p_{(a,1)}$, then the flow should decrease on another path $p_{(a,2)}$ (where $p_{(a,2)}$ uses some hub h_2). Now, the flow on some path $p_{(c,1)}$, that also uses hub h_2, has to decrease for the amount of flow in the hub to stay the same. Now we can continue this argument until the flows on $p_{(a,1)}$ - $p_{(a,2)}$ - $p_{(c,1)}$ - ... - $p_{(b,1)}$ are adjusted in a + - − - + - ... - − fashion, such that the total flow of every commodity is the same, and no hub capacity is violated. This results in a new feasible solution that has a different total profit.

This observation (3-4.2) is an intuitive example of the main argument in Proposition 3-4.3.

Proposition 3-4.3. *Assume that for a solution s, C^s is the set of commodities for which the flow is split over multiple paths, and H^s is the set of open hubs. Then for any optimal solution, we have that $|C^s| \leq |H^s|$.*

Proof. We will prove the proposition by contradiction. We will show that from any solution s such that the proposition is not true, i.e. $|C^s| > |H^s|$, we can construct a feasible solution

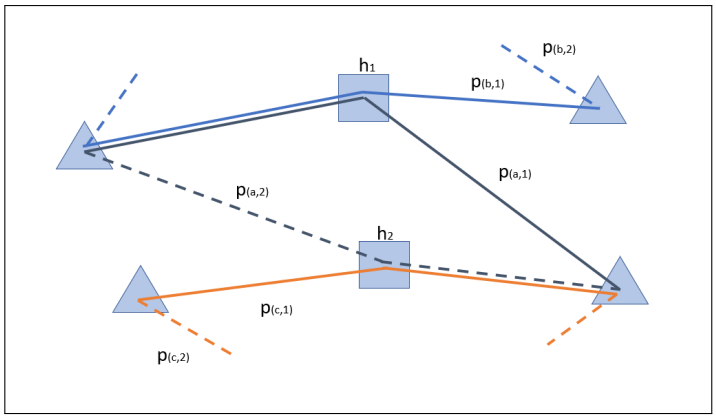


Figure 3-2: The splitting of flow over multiple paths.

that is better, which contradicts optimality. We do this by keeping the revenues the same and lowering the costs while retaining feasibility.

First, we note that for a commodity of which the flow is split over multiple paths, it holds that if one of those paths is the direct path, sending all flow over this path will create a better solution. This is because in reality, the variable costs of this path will always be lower than any path that uses a hub. Thus, we can create a better solution than s if it has such a commodity. Therefore, we continue assuming that in s all used paths of the commodities of which the flow is split, are not direct paths (i.e. they use at least one hub).

Now we will show that for s, a new solution can be constructed, which has a better objective value. For this, we let P^s be the set of paths that have positive flow in s and belong to one of the commodities in C^s. We assign a variable x_p to every path $p \in P^s$, which represents the change in flow over that path compared to the flow it has in s. Assume these variables are the elements of vector $\mathbf{x}$. For the hubs, the change in flow can now be described as

$$\sum_{\substack{p | p \in P^s \text{ and} \\ p \text{ uses } h}} x_p \qquad \forall h \in H^s. \tag{3-10}$$

And for the commodities, the change in flow can be described as

$$\sum_{\substack{p | p \in P^s \text{ and} \\ p \text{ belongs to } c}} x_p \qquad \forall c \in C^s. \tag{3-11}$$

Now, if we find a non-trivial solution $\mathbf{x}$ such that (3-10) and (3-11) are equal to zero, then we have found a different feasible solution with the same revenues as s and a different cost (because the flows over the paths are different and in reality the variable cost of every path will be different). These equalities form a system of equations, and we get the following inequality for the number of variables n, and the number equations m.

$$n = |P^s| \geq |C^s| + |C^s| > |C^s| + |H^s| = m$$

The first inequality is true because every commodity in C^s has at least two paths with positive flow, and the second inequality is true because the proposition is not true for s. This shows we

have a homogeneous system of linear equations where the number of variables is greater than the number of equations, which means a non-trivial solution $\bar{\mathbf{x}}$ exists. This solution shows how the flow on every path can be changed. Because the objective function is linear, applying either the change $\bar{\mathbf{x}}$ or the change $-\bar{\mathbf{x}}$ will result in a feasible solution with a better objective value. Which is the contradiction we were after, and this proves the proposition. □

In reality, the splitting of flows is a negligible effect since the quantities of parcels that are involved in this are so small that they can be easily handled by the hubs, even if they are theoretically already handling all they can. Here it is also worth noting that the way reality is modeled is inevitably a coarse approximation. For example, the parcels are taken as one amount per day, while in practice they are spread over multiple batches during the day. This means the hub capacity is not as strict in reality as the model formulation might suggest. Of course, mathematically this does affect the solution. This is covered in Subsection 3-4-4. The balancing constraints cannot enforce multiple path-choices since in this case as well, one option will be the cheapest one. That is, either choosing a path that reduces the need for balancing, or choosing to reposition a vehicle, is cheapest for the objective function.

The non-linear model formulation contains an "on/off"-constraint (3-6) for every potential arc. These regulate the possibility of flow over an arc and can only be tight when the corresponding indicator variable is equal to 0. Here, we have chosen a formulation that has a constraint for every arc on a path. Why this formulation is chosen and what the other options are in this case is covered in Subsection 3-4-6 on possible reformulations of the model.

Additional Parameters

$\dot{x}_{\max}(k)$	maximum market share of commodity k,
$\dot{x}_{\max}(p)$	maximum market share of commodity k such that k belongs to path p,
a_l	offset of linearization function l,
b_l	gradient of linearization function l.

Additional decision variables

y_k	New revenue variable for commodity k,
$\dot{x}_p$	market share of path p.

Model formulation

$$\max_{y,\dot{x},\widehat{z},\breve{y}} \quad \sum_{k\in K}(y_k - \sum_{p\in P^{(k)}} \dot{x}_p \cdot (f_k^w c_p^w + f_k^{pcs} c_p^{pcs})) - \sum_{s\in S} \widehat{c}_s \widehat{z}_s - \sum_{(d_1,d_2)\in(D\times D)} \breve{c}_{(d_1,d_2)} \breve{y}_{(d_1,d_2)} \tag{3-12}$$

$$s.t. \quad \dot{x}_p \geq 0 \qquad \forall p \in P \tag{3-13}$$

$$\sum_{p\in P^{(k)}} \dot{x}_p \leq \dot{x}_{\max}(k) \qquad \forall k \in K \tag{3-14}$$

$$y_k \leq a_l + b_l \sum_{p\in P^{(k)}} \dot{x}_p \qquad \forall k \in K, \forall l \in L \tag{3-15}$$

$$\widehat{z}_s \in \{0,1\} \qquad \forall s \in S \tag{3-16}$$

$$\dot{x}_p \leq \widehat{z}_s \cdot \dot{x}_{\max}(p) \qquad \forall p \in P, \forall s \in S^{(p)} \tag{3-17}$$

$$\sum_{k\in K} \sum_{p\in P^{(k)}\cap P^{(h)}} {f_k}^{pi} \dot{x}_p \leq C_h \qquad \forall h \in H \tag{3-18}$$

$$\breve{y}_{(d_1,d_1)} \geq 0 \qquad \forall (d_1,d_2) \in (D\times D) \tag{3-19}$$

$$\sum_{p\in P^{O(d)}} {f_p}^w \dot{x}_p - \sum_{p\in P^{D(d)}} {f_p}^w \dot{x}_p = \sum_{d_1\in D} \breve{y}_{(d,d_2)} - \sum_{d_1\in D} \breve{y}_{(d_1,d)} \qquad \forall d \in D \tag{3-20}$$

In the objective function, the total profit is maximized. The revenue is represented by the new variable y_k, which is dependent on the market share $\sum_{p\in P^{(k)}} \dot{x}_p$ through the linearization constraints (3-15). The costs are composed of the variable costs over the paths (product of market share, total flow, and unit cost), the fixed costs of the arcs, and the balancing costs, which are dependent on the market shares through Constraints (3-20). Constraints (3-13) and (3-14) bound the market shares. Constraints (3-15) are the linearization constraint as they are explained in Section (3-4-2). Constraints (3-16) set the binary range for the arc variables, and Constraints (3-17) ensure there can only be a positive market share if all arcs on that path are open. The multiplication with the maximum market share in this constraint strengthens the formulation. In Constraints (3-18), the flow is summed over all paths that use a certain hub, and this value is ensured to be at most the hub's capacity. Constraints (3-19) and (3-20) ensure an imbalance in flow between depots is penalized.

3-4-3 The approach

For finding a solution to the above formulated model, several strategies are possible. We have chosen to use a general purpose MILP solver, combined with a pre-selecting procedure for the arcs and the paths. An advantage of doing this is that parts of the results are also applicable to other situations where a general purpose MILP solver is used.

The MILP solvers and the algorithm (B&C) they use is further explained in Subsection 2-3-2. For the solver two options were tested on some early versions of the model, namely CPLEX and GUROBI. In the section on the background of the software that is used (Section 3-4-2) these solvers are discussed further. Based on the solution time it needs and the possible LP-gap, the tests revealed a clear preference for GUROBI. This is confirmed by benchmark tests that compare the two, such as is done by Mittelmann (2018).

The GUROBI solver can be tuned to get better results for the specific model that it is run on. This is done by using the GUROBI tuning tool, which performs part of the solution process multiple times with different parameters setting and searches the very large space of parameter options for the ones that improve the performance the most. It improves performance in a heuristic way, and can therefore result in significant variance in performance. This variation is also observed in practice where the tool does not improve the solution time at all in some cases, and in some other cases the improvement is up to 30%. For most of the tested cases the improvement in solution time is around 5%.

Linearization Choosing the linear approximations is done in a way that ensures the error of the approximation is as small as possible. This is achieved by minimizing the average distance between the quadratic curve and the linear functions, on the ranges that they are active. The minimization model that is used for this minimizes the square of the area (as depicted in Figure 3-3) between the functions by determining the optimal gradient and offset of the linear functions. An advantage of this method, is that for intervals of the function that are curved, the method will automatically choose more lines than on intervals where the function is "more linear". To ensure that a certain error is achieved, an option is to run the model multiple times with an increasing number of linear functions and track the difference between the linearized and quadratic objective functions for the optimal solution. But as most actual networks will be large (and thus the solution time will be long), this is not possible for every situation. That is why in the next section, we have analyzed the error for multiple amounts of linear functions, so an appropriate amount of linear functions can be chosen in advance.

Using one line for the linearization A consequence of the linearization approach as it is described above, is that for all commodities, the combination of the market share and revenue that is chosen in the optimal solution, lies on an intersection of lines. This is true for all cases that were tested for this section or for the sensitivity analysis in the next section. The reason that this happens, is that if there are no restrictions on the market share of a commodity, we can view the sub-problem of choosing the optimal market share for an isolated commodity as an optimization over a piecewise linear function, for which the optimum always lies in a vertex. A result of this, is that if the approximation of the non-linear function is done by only using one line, the problem of choosing market shares changes into the problem where for every commodity there is one fixed market share and price, and the model can choose to open or close a commodity. We will call this principle quasi-binary, since it is not necessary

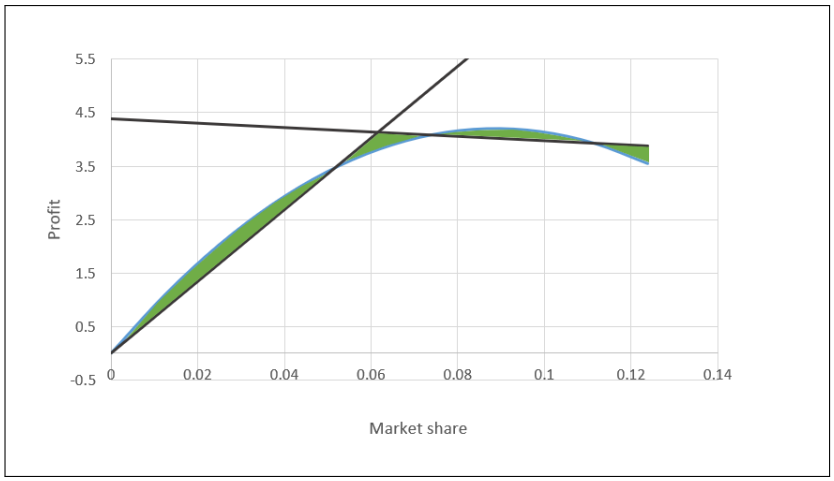


Figure 3-3: Minimization of the linearization error.

to not enforce the binary choice by constraints.
In comparison to using multiple lines for the linearization, using only one line results in a short solution time, but also in a large error, i.e. the difference between the objective value of the optimal solution of the linearized model compared to the one of the non-linear model is large. However, this can be significantly limited by choosing the linear function in a smart way. For this we note that, for most commodities, the market shares that are chosen, are the ones that would also be chosen in the sub-problem of choosing the optimal market share for an isolated commodity if only the revenues and variable costs are included. If we choose the linearization, such that the model either chooses the optimal market share of the non-linear model or chooses a market share of zero, the error will be small. This is shown in Figure 3-4.

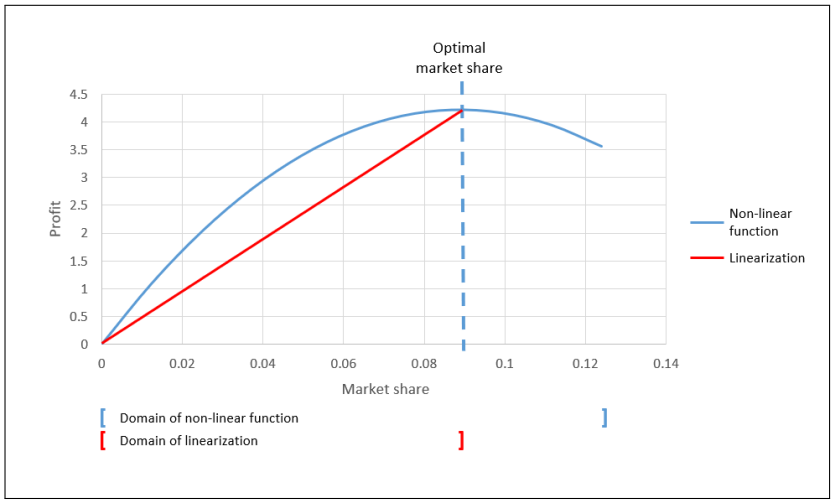


Figure 3-4: The best way to choose the linearization of a commodity if one line is used, and only the revenue and variable costs are taken into account.

Pre-selection of arcs In every optimal solution only a subset of the arcs will be open. If we know in advance which arcs are in that set we can speed up the solution process significantly by only using these as potential arcs. Of course, in practice it is impossible to know the exact set of arcs, but we can approximate it by looking at the solution to the LP-relaxation. This is fast to solve and gives us a fractional solution for the arc choices. Now if in the MILP model, we only use the arcs that have a positive market share, we get a decrease in solution time, and we pay for it by an added error. The trade-off between these is covered in the next subsection.

Pre-selection of paths For the path-choice, a similar statement holds, but using the same technique as for the arcs is not possible as the LP solution returns one path for the majority of commodities, and for most commodities this is not the path that is chosen in the optimal solution of the MILP model. Excluding some paths from the potential set could nonetheless improve the performance. That is why the following procedure is used: First the potential arcs $P_{(0)}$ (either all possible arcs, or a subset determined by the above-mentioned pre-selection of arcs) are all fixed open. This turns the MILP model into an LP model as the arc variables were the only integer variables. Then we start an iteration process where, for iteration i such that $1 \leq i \leq n$:

- The LP model is run, where the set of potential paths in this iteration is

$$P = P_{(0)} \setminus \bigcup_{j<i} P^{(j)},$$

  the solution to this model has one or more paths with positive market share for every commodity, and we denote these paths by $P^{(i)}$.

The value of n depends on the required error and solution time. The set of paths that is used in the actual MILP is then

$$P^* = \bigcup_i P^{(i)},$$

In the first iteration, the first-choice paths for every commodity are discovered, in the second cycle the second-choice paths, and so forth. The resulting error, and the consequences for the solution time are covered in the next section.

Using GUROBI as a heuristic For larger networks the combination of methods that was just described will likely not be possible to solve the problem to optimality. In this case an option is to stop the solution process of GUROBI before the found solution is proven to be the optimum. This way the method serves as a heuristic that generates a solution and an LP-gap, which is a bound on how far the objective value of the solution can maximally be from that of the optimal solution. This is a method that is used by ORTEC frequently in solving network optimization MILP models. Using this is also motivated by the fact that for many difficult optimization problems the built-in heuristics of GUROBI find good quality, or even optimal, solutions early in the solution process, but the solver has great difficulty in

proving how good the solutions actually are. This is something that is inherent to NP-Hard problems. The tuning of the solver parameters to this specific goal is something that can enhance the outcomes. The evolution over time of the LP-gap, which determines when it is feasible to stop the solution process, is given at the end of the subsection below.

3-4-4 Error

A solution that is found using the approach as described above, differs in multiple ways from the optimal solution of the model in its original quadratic formulation. This error is what is covered in this subsection. The different paragraphs in this section cover the error of the corresponding part of the approach, as they are discussed in the previous subsection.

Introductory remarks It should be noted that the way the problem is modeled is necessarily a coarse simplification, e.g. the price-market share relation will be very hard to compute accurately in practice, and that the method is intended for tactical/strategical purposes. This is why achieving a very small error is not so important. Also, the results in this subsection are attained by using a fairly small network, and the unpredictability of some results make translating results to larger networks not straightforward. However, most parts of this subsection are tested on two input sets that have different values for the total flow and the prices. This verifies that these parts do not significantly change the results. They are generated using the approach that is explained in Section 3-5. And they have low variance for both the prices and the total flows, and they have hub capacities that result in a relatively small solution time, which ensures a lot of runs can be done.

Linearization Part of the error is caused by using a linearized approximation of the objective function. As briefly discussed before, this error is caused by two contributions. The first is when a market share has been chosen, the revenue is different in the quadratic function compared to the linear approximation. This error can be prevented by plugging the market shares from the solution of the linearized model into the quadratic functions and calculating a new total profit using these revenues. The second contribution is that a different market share is chosen because the objective function and constraints are different in the linearized model. This influences the solution that is found in multiple ways. One of them is the following possible chain of consequences: a different market share is chosen, which leads to a different flow, to a different chosen path, to a different set of chosen arcs, to a different total cost, and eventually to a different total profit. Unfortunately mathematically bounding the total error a priori is not possible since the effect on the objective is too unpredictable.
As for the number of linear functions that are used, we can assume that the error will be smaller when more of them are used to approximate the quadratic function. To check how large the error of a found solution actually is, we need the optimal solution to the quadratic model. To find that solution, we have tried several approaches, the most promising of which was reformulating the problem as a SOCP-problem. This is a class of convex problems that

is more general than LP-problems, but whose problems can still be solved in polynomial time by interior point methods. The SOCP-model was solved using GUROBI's specific methods, and the solution time was too large for this method to be of practical value. So we concluded that the precise error cannot be found, but it *can* be approximated.
For this we used an increasing number of lines to approximate the quadratic function and we looked at how the total profit evolves. We distinguish between two versions of the objective value; the first is the objective value of the model with the linearized revenue functions, the second is calculated by plugging the market shares from the solution of the linearized model into the quadratic functions for the revenue, and then by calculating a new total profit using these values. From now on we will call the first one the *linearized value*, and the second one the *quadratic value*. We also recall that for all commodities the combination of the market share and revenue that is chosen in the optimal solution, lies on an intersection of lines. In Figure 3-5-a we see that both the quadratic and the linearized values converge as more lines are used for the approximation. In Figure 3-5-b we see that the difference between the two values converges with a constant rate. This makes sense as with every extra line that is used, the intersections of the lines are closer to the quadratic function. These figures show that we can assume that the optimal solution to the quadratic model is very close to the quadratic value when 50 lines are used.

In Figure 3-6-a we compare the approximation that uses 50 lines to the values that are found for different numbers of lines. We see that when less than 18 lines are used, the errors behave erratic, and that after this point the values behave more smoothly. This makes sense as when only a few lines are used and a line is added, the distance in the two-dimensional plane between the old and the new intersections is relatively large, and the optimal solution might "jump to a different intersection". This seems to also be the case for the outlier when 3 lines are used. Another interesting aspect is the fact that the linearized value is always relatively close to the optimal solution, this is even more obvious in Figure 3-5-a. Here, the fact that a market share is chosen that is different from the optimal solution of the quadratic problem, which makes the objective value too low, is compensated by the fact that the intersections of the lines are above the quadratic function, which makes the revenues, and thus the objective value, too high.
The choice for which amount of lines should be used is dependent on the solution time at hand(Figure 3-6-b) and on the context in which the model is solved. If an error below 1% is sufficient, using only 4 or 5 lines and the linearized value is possible, while lowers errors are best achieved by using more lines and the quadratic value. We suspect that the solution time is irregular because the solver is lead in a better or worse direction in the B&B-tree. We know from the log-files of the solver that the time it takes to find good quality solutions is not the cause of this, but the location of the intersections of the lines *can* influence the way in which the solver heuristically chooses which branch to explore next.

Pre-selection of paths To establish the error of the pre-selection of paths, we first look at the results (Figure 3-7) when the n cheapest paths are used to find an approximation of

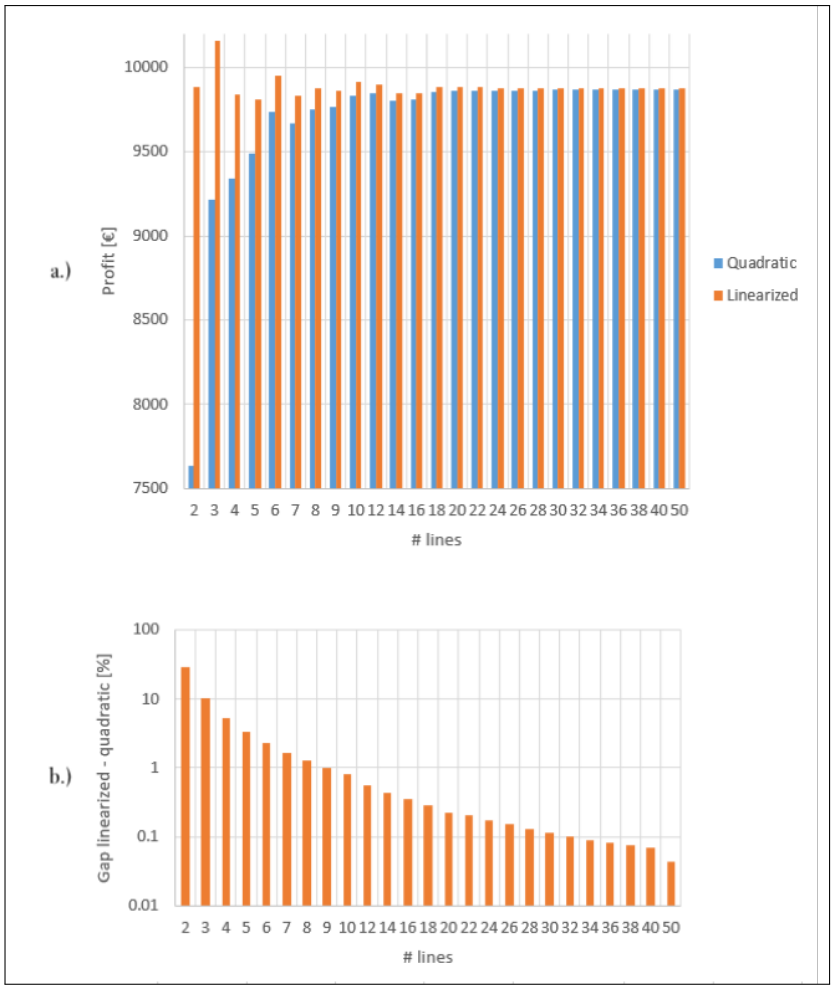


Figure 3-5: Convergence of the error as more lines are used for the linearization.
a.) Convergence of the linearization error.
b.) Convergence of the difference between the quadratic and linearized values.

the optimal solution when an infinite number of paths would be used. The objective value converges as more paths are included, and the value that it converges to, is used to determine the error when the previously described pre-selection procedure is run a preset number of times. These results are shown in Figure 3-8. What most stands out is the fact that when the procedure is run four times, the solution time is about 430% more, than when the procedure is run five times. Both these solution times are vastly different from the general trend in the graph, and also from what one would expect from the development of the error. Looking at the log-files, we conclude that the solver is quicker in finding better integer solutions during the B&B-process. Apparently the fifth-choice path makes the decisions to be made a lot easier. We suspect this is a network-specific effect. This also shows that it is not the number of paths that is important for the solution time, but which specific paths are in the potential set. A lesson to learn from this is that if the solution time of a similar model is long, both including more or less paths can decrease solution time. What is also interesting in the graph

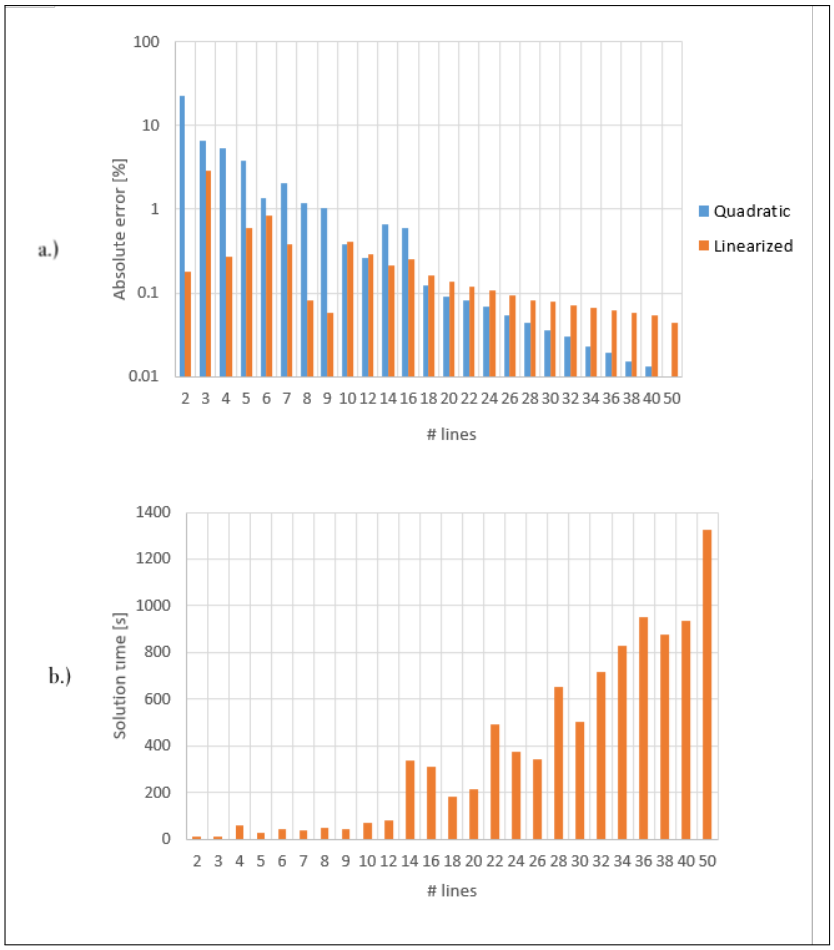


Figure 3-6: The error as a function of the number of lines that are used for the linearization.
a.) Absolute error of the linearization approach.
b.) Solution time of the linearization approach.

is the difference in solution time between using two paths per commodity and using three paths per commodity. This is to be expected since for all commodities it holds that the first path is the direct path, and the second and third path go through a hub. This makes the decision a lot harder when there are three paths to choose from, as the second and third path costs are more similar, and they influence other commodities.

Path relaxation A possible consequence of the relaxation of the constraint that only one path can be chosen, is that the objective value is too high. And, as shown in Proposition 3-4.3, the number of commodities for which multiple paths are chosen, is at most the number of hubs. This leads us to Observation 3-4.4.

Observation 3-4.4. When there is a commodity for which multiple paths are selected, removing the market share on all but one of these paths gives a feasible solution to the non-relaxed problem. Therefore, this gives an upper bound on the gap between the optimal values

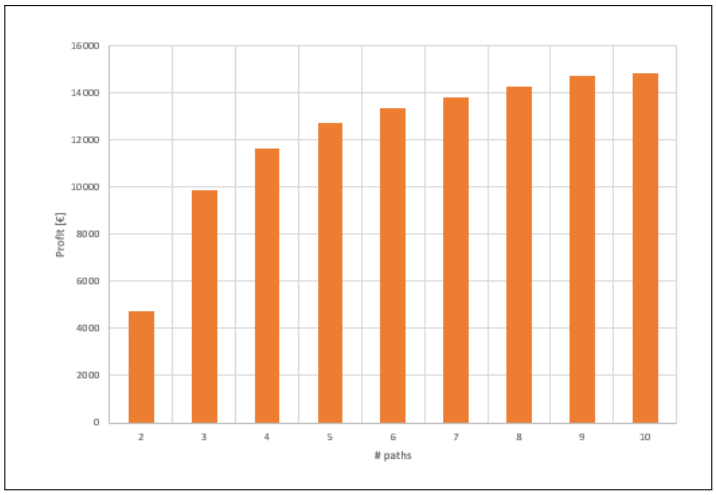


Figure 3-7: Objective value as a function of the amount of paths that are included in the model.

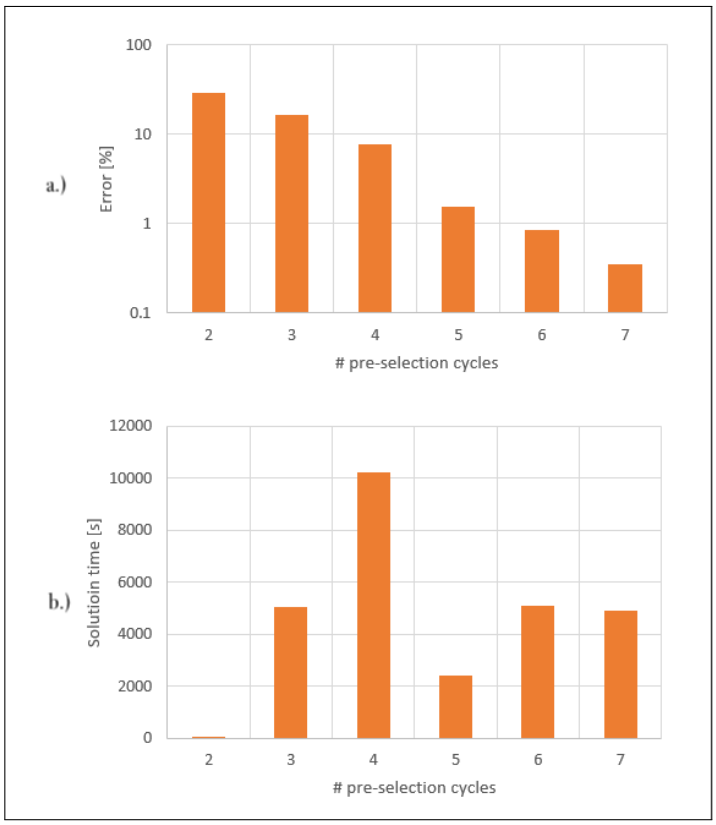


Figure 3-8: Solution time and error as a function of the number of times the pre-selection cycle is run.

of the relaxed model and the non-relaxed model. To make this bound as tight as possible the path that results in the highest profit should *not* be removed. The largest profit among these paths is at least $\frac{1}{\hat{p}^c} \cdot \rho^c$ where $\hat{p}^c$ is the number of possible paths for the commodity

and ρ^c is the maximum profit of the commodity. Therefore, theoretically the error is at most $\frac{1}{\hat{p}^{max}} \cdot \rho^{max} \cdot |H|$, where $\hat{p}^{max}$ is the number of possible paths for *any* commodity, ρ^{max} is the maximum profit of *any* commodity, and H is the set of all hubs.

The actual error has been tested on a variety of different cases where, in addition to the total flow and prices, the number of lines of the linearization, the number of included paths, and the hub capacities were varied. This showed that there are almost no cases in which multiple paths are selected. The error is 0% in almost all cases, and less than 1% in the rest of the cases. The improvement in solution time is between 7% and 75%, and most cases are above 50%. The log-files of the solver showed that this is partly caused by the pre-solve procedure that takes less time, and partly by the shorter time it takes to heuristically find the optimal integer solution.

Pre-selection of arcs Another part of the approach that causes the solution to be lower than the optimal solution to the quadratic model is the pre-selection of arcs. Bounding this error a priori is not possible, but running the model with multiple different input sets shows that the error is between 0% and 4%, and that the improvement in solution time is between 80% and 99%. This improvement is caused by the removal of (integer) arc-variables from the model. The uncertainty of the error makes pre-selection of the arcs a something that us not possible to use in many situations.

Using GUROBI as a heuristic To see whether this is a viable option for cases where the time needed for finding the optimal solution is *too* long, we will analyze the quality of the solutions that are found during the solution process. This is done for a case that takes a long, but still practical amount, of time to solve. For this, a case is created that has relatively low hub capacities, which makes the model hard to solve as is further explained in Subsection 3-5-1. During the process, the actual error is not known because the optimal solution is not yet known. The LP-gap, which is the difference between the best solution that was found so far and the solution to the LP-model after the adding of cuts, is then the best indicator of the quality of the solution. This will be used in practice to determine when to stop the solution process. The relationship between these two values can be seen in Figure 3-9.

We see that the solver finds good quality solutions early in the process, and after 2700 seconds, the optimal solution is found (the part of the process where the y-axis is zero is not plotted due to the logarithmic scale). However, it takes a long time to reduce the LP-gap to zero, so it takes a long time to prove the found solutions are good or optimal. This means stopping the solver before it guarantees optimality is a good option for situations where the guarantee is not necessary. Stopping criteria that will work well, judging by the graph, is stopping the solver when for a certain amount of time no better solution is found, or when the improvement between one or more successive solutions is below a certain threshold. Furthermore, settings for GUROBI are available that either lead the solver towards finding good solutions early in the process, or proving optimality as quickly as possible. If the approach of using GUROBI

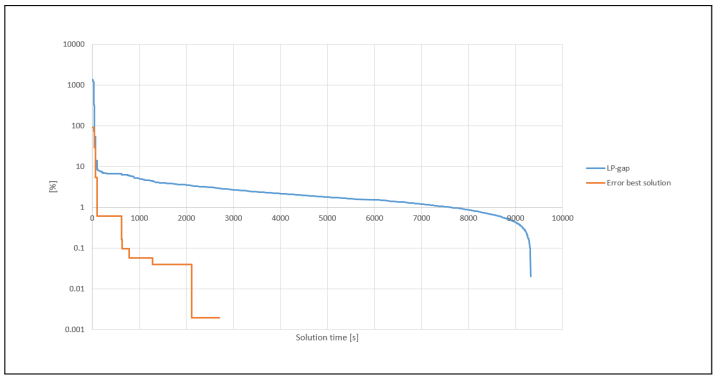


Figure 3-9: Development of the LP-gap, and the error of the best solution that is found thus far (i.e. the difference between the solution and the optimal solution). For 2700 seconds and later, the error of the best solution is 0%, this is not plotted due to the logarithmic scale.

as a heuristic is used, changing these settings could improve performance.

Summary In Table 3-2, we have summarized the findings of this section. In the leftmost column the *required* total error is shown. This is the error that one requires for the objective value of the solution, which in practice will depend on the context in which the problem is solved. Then there are two entries in the middle cells of the table. The first entry is a factor that shows how large the total solution time is, which is normalized with respect to the row in the table that shows the 5% case. Between the brackets the setting for that part of the solution approach is given. Combined over all parts, these settings will result in the required error. In the rightmost column, the total factor for the solution time is given. This is a multiplication of the factors of all the parts , the accurateness of this value has not been tested, and is thus an estimation ($\approx$).

Table 3-2: Consequences of the different parts of the solution approach. Entries consist of the normalized solution time, and between brackets is the setting for the part that is shown in the top row.

Error	Linearization	Path relaxation	Path pre-select	Arc pre-select	GUROBI as heuristic	Total
5%	1 (5)	1 (yes)	1 (5)	1 (yes)	1 (yes)	1
1%	2 (8)	1 (yes)	2.1 (7)	12 (no)	6 (yes)	$\approx$302
0.1%	27 (40)	1 (yes)	2.7 (no)	12 (no)	12 (yes)	$\approx$16,796

This combination of settings is one out of many possibilities, and a different combination of settings may also lead to the required error. These settings were chosen because they have the lowest expected total solution time. Also, as these parts all have an uncertainty to them, the results in the table are a best estimate. This means the total required error cannot be guaranteed, but will most likely be met. Furthermore, the value for the total solution time assumes that multiplying the factors gives a good approximation of the total time. Although

this has been verified on a few combinations, it has not been extensively tested and can thus only be interpreted as an indication of the performance.
There are two settings that need further elaboration. The first is the path relaxation, which is always allowed since the error it causes is positive, while the rest of the settings cause negative errors compared to the actual optimal solution. It is very small in comparison to the negative errors. And so it will only decrease the total error. The second is using GUROBI as a heuristic. We have looked at the time when the actual desired error is reached, of course, in reality this error is not known during the solution process and so a suitable stopping criterion needs to be found that leads to the correct approximate error. This stopping criterion will not stop the solver at the exact correct time and so the actual solution time will increase some more, which we have accounted for with an addition to the solution time of 20%.
We see from the table that in almost no situation, requiring an error of 0.1% will be possible, as the solution time would be too large. However, this will probably also not be needed when using the model in a tactical/strategic context.

3-4-5 Analysis of the solution process

In this subsection, we analyze the solution process of the MILP model, the theoretical explanation of which is covered in Subsection 2-3-2. The effects of variations in the input data on the solution time are not covered here, but in Section 3-5. Furthermore, the tuning of the GUROBI solver is left out, since this makes the results more unpredictable. We also want to mention that the GUROBI solver is somewhat of a black-box algorithm that can behave in unexpected ways. Detailed explanations about its behavior are therefore not possible in most cases, although most general behavior can be explained based on the basic principles of the algorithm. This is true for all B&B and B&C based algorithms, especially when they are used to solve NP-Hard problems.

The constraints that add the most complexity to the problem are the hub capacity constraints, this is extensively covered in Section 3-5-1. The consequences of these constraints on the solution are that non-maximal profits and/or different paths are chosen.
The balancing constraints add about 200% to the solution time. Furthermore, the consequence for the optimal solution is that (for realistic costs) the balancing is completely done by adjusting market shares and choosing different paths, so the repositioning of vehicles is avoided.

For a more detailed analysis of the solution process, we will look at two different cases:

- Case *a* has hub capacities such that in the optimal solution half of the hubs handle a flow that is equal to their capacity.
- In Case *b* this holds for *all* hubs

The reason we make this distinction is that the difference in solution time between these cases is vastly different (more on this in Subsection 3-5-1), and they both represent realistic situations. The other input data is generated such that it is an representative and predictable

case.

LP-relaxation The first aspect that we will analyze is the solution to the LP-relaxation. If in this solution, the values of the variables that are required to be integral, are *almost* integral, the model is likely easy to solve as it is already close to the optimal solution to the MILP model. The variables that are required to be integral are the binary arc decision variables. From the arcs that these variables represent, a subset will have the value 1 in the optimal solution to the MILP model, and a subset will have the value 0. In Figure 3-10, we see how the fractional variables are distributed.

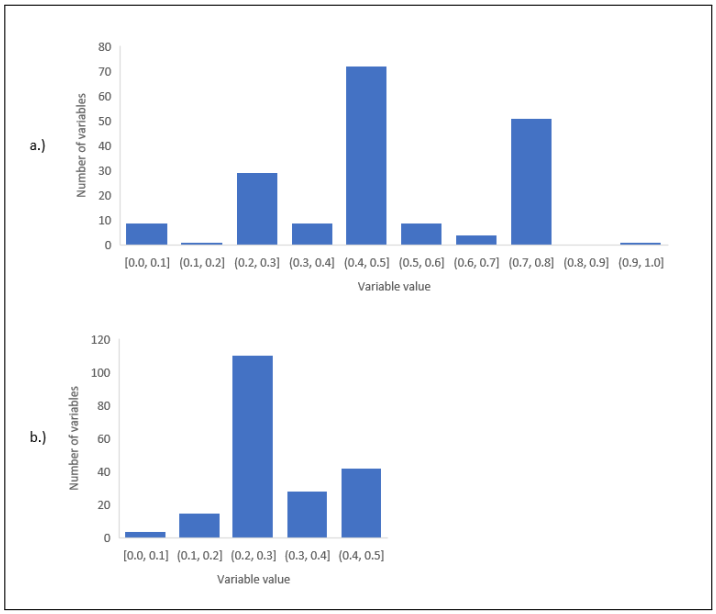


Figure 3-10: Distribution of the fractional variables in the optimal solution of the LP-relaxation for Cases *a* and *b*.

The reason that for Case b there are no fractional values above 0.5, is that the hub capacities limit the flow over the arcs more than for Case a. Furthermore, in Case a,, the fractional values are a lot closer to binary values than for Case b, and as we have seen, this results in a lower LP-gap, and a shorter solution time.

Pre-solve During the pre-solve, the solver tries to detect special characteristics and implement certain changes in the input that will most likely lead to a better performance of the solution process. More information on this process is given in Section 2-1. This results in the removal of about 27% of constraints and 31% of variables for Case a and about 16% of constraints and 21% of variables for Case b. Examples of these are constraints that are also implied by a combination of other constraints, and variables that can logically not be in any optimal solution. If the pre-solve is turned off the solution time increases 72% for Case a and 78% for Case b.

Root-node In the root-node of the B&C-algorithm the most important step is the adding of cutting planes, which are constraints that make the LP-model stronger without cutting off integer solutions. After this process the LP-gap is respectively 0% and 13% for the two cases. This means that for Case *a*, the part of the feasible region in which the optimal solution is located, is cut until the integer hull remains. Then the integer points are perfectly described by the constraints, and the optimal solution to the LP-relaxation is the same as the optimal solution to the MILP model, and so the entire solution process is completed. For Case *b*, this is apparently not possible using only the kind of cuts that are implemented in the solver, or this would take too much time. In Table 3-3 we see how many cuts are added for the two cases. The exact reasons that these kinds of cuts are added in their specific amounts are unknown because the adding of cuts is done in an undisclosed heuristic way. Nonetheless, we can say it is logical that the number of flow cover cuts is the largest since we are dealing with a flow problem with on/off constraints. Furthermore, finding out which inequalities are actually added to the model is unfortunately not possible. However, we suspect in Case *b* it is less obvious where in the feasible region the optimal solution lies, and so a lot more cuts are added to help the solution process.

Table 3-3: Number of cuts that are added during root-node-processing.

	Number of added cuts	
	Case *a*	Case *b*
Implied bound	5	13
Gomory Mixed Integer Cut	6	53
Cover Cut	5	-
Clique Cut	2	-
MIR Cut	17	413
Flow cover Cut	213	2155

The tree Next we look at the B&C process that starts after the root-node, which is only applicable to Case *b*. The evolution of the LP-gap and the difference between the best found solution and the optimal solution is given in Figure 3-11. As we have seen in the previous subsection good quality solutions are found quickly, but proving that they are good takes a relatively long time. This means a lot of branching and cutting has to be performed for the LP-objective to be decreased, but that the heuristics of the solver do a good job in finding integer solutions from the LP-solutions.

3-4-6 Formulation

The consequences of different formulations of a model on the performance of the solution process is often difficult to predict. The best way to evaluate it is to implement changes and analyze the results. These different formulations are what is covered in this subsection, these deliberations result in the formulation at the beginning of this section.

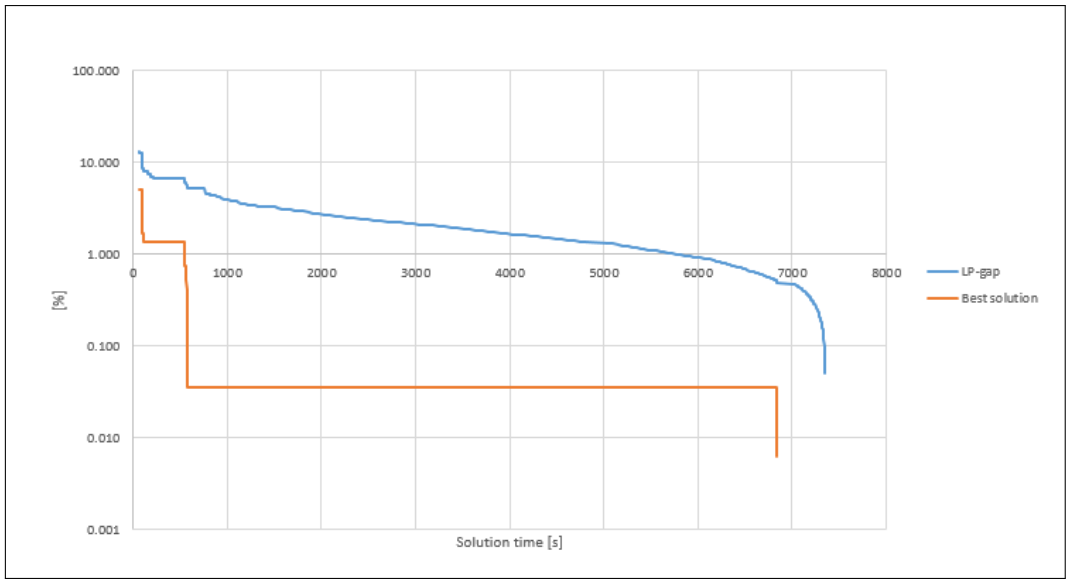


Figure 3-11: Development of the LP-gap, and the error of the best solution that was found thus far (i.e. the difference between the solution and the optimal solution). For 6800 seconds and later, the error of the best solution is 0%, this is not plotted due to the logarithmic scale.

"On/off" constraints First we look at the possibilities for implementing the "on/off" constraints (3-6). These are the constraints that regulate the possibility of flow on the arcs. The first option is using the built-in indicator constraints of AIMMS. These are only imposed to hold if a certain variable has a certain value, and are dealt with by AIMMS in an unspecified way. Testing has revealed that using indicator constraints is slow compared to some of the other options in this subsection, which is probably the result of the extra variables that are introduced.
Next we look at Hijazi et al. (2012), who list a few options and analyze them from the perspective of MINLP problems (a special case of which is MILP). From this we get the second option, which is transforming the constraints to

$$\dot{x}_p \cdot (1 - z_s) \leq 0 \qquad \forall s \in S, \forall p \in P^{(s)}$$

Here $s \in S$ is an arc, $p \in P^{(s)}$ is a path from the set of all paths that use the arc, $\dot{x}_p$ is the market share of the path, and z_s is 1 if arc s is open and 0 if it is closed. The consequence of this formulation is that even the LP relaxation is non-linear and possibly non-convex. This is why this approach is also not suitable.
A third option that is mentioned in Hijazi et al. (2012) is using disjunctive programming (Balas (1979)), which is an alternative modeling approach to MILP. This has been proven to be a successful modeling framework for problems that include discrete decisions (Hijazi et al. (2012)). In this method the feasible region is divided into polytopes in which an arc variable is 1 and polytopes in which an arc variable is 0. The new feasible region is then taken as the convex hull of the union of these polytopes. If the integrality constraints on the arcs are dropped in this model, good upper bounds are typically found on the original formulation.

Using this approach seems like a promising approach, but it is not used in this thesis since implementing this would consume a lot of time, and this field is not an area of expertise of the university group for which this thesis is written.
The fourth option introduces a constraint for every arc $s \in S$ on a path $p \in P^{(s)}$ and links the market share $\dot{x}_p$ and the arc z_s variable. It is possible because the upper bound of the market share is lower than 1 (which is also the maximum of the arc variable). From this point on we will refer to these constraints as one-by-one-constraints:

$$\dot{x}_p \leq z_s \qquad \forall s \in S, \forall p \in P^{(s)}. \tag{3-21}$$

The fifth option contains the sum of the market share of all paths that use the arc and then uses a Big-M constraint. These we will refer to by the sum-constraints:

$$\sum_{p \in P^{(s)}} \dot{x}_p \leq z_s \cdot M \qquad \forall s \in S. \tag{3-22}$$

This method is well-known and has a few implications such as potential problems with numerical stability and lower quality of the LP-solution. However, these problems are mostly caused by choosing a very large M, which in our situation can be small, i.e. $x_{\max} \cdot |P^{(s)}|$ where $P^{(s)}$ is the set of paths that use arc s. To compare these last two options we first look at the LP-gaps, for which we get the following result.

Proposition 3-4.5. *Assume that $M = \dot{x}_{\max} \cdot |P^{(s)}|$, then the sum-constraints are strictly stronger than the one-by-one-constraints.*

Proof. First we show that for any arc $s \in S$ and any $(\dot{\boldsymbol{x}}, z_s) \in ([0, \dot{x}_{\max}]^n \times [0,1])$, where $n = |P^{(s)}|$, it holds that

$$\sum_{p \in P^{(s)}} \dot{x}_p \leq z_s \cdot M \qquad \Rightarrow \qquad \dot{x}_p \leq z_s \qquad \forall P^{(s)} \ .$$

Suppose we have an arc s and a path q that uses s, such that $\dot{x}_q > z_s$. Then we get that:

$$\sum_{p \in P^{(s)}} \dot{x}_p = \tag{3-23}$$

$$\dot{x}_q + \sum_{p \in P^{(s)} \setminus q} \dot{x}_p > z_s + \sum_{p \in P^{(s)} \setminus q} \dot{x}_p \tag{3-24}$$

$$= z_s + \sum_{\substack{p \in P^{(s)} \setminus q \\ |\dot{x}_p \leq z_s}} \dot{x}_p + \sum_{\substack{p \in P^{(s)} \setminus q \\ |\dot{x}_p > z_s}} (z_s + (\dot{x}_p - z_s)) \tag{3-25}$$

$$\geq z_s \cdot |P^{(s)}| + \sum_{\substack{p \in P^{(s)} \setminus q \\ |\dot{x}_p > z_s}} (\dot{x}_p - z_s) \tag{3-26}$$

$$> z_s \cdot |P^{(s)}| \tag{3-27}$$

$$> z_s \cdot |P^{(s)}| \cdot \dot{x}_{\max} \tag{3-28}$$

$$= z_s \cdot M \tag{3-29}$$

This proves that the sum-constraints are stronger than the one-by-one-constraints. To show that this inequality is strict we show that there exists an $(\boldsymbol{x}, z_s) \in ([0, x_{\max}]^n \times [0,1])$ such that the one-by-one-constraints are satisfied, but the sum-constraints are not. For this we take $\dot{x}_{\max}$ both as the value for z_s as for the value at every position in the vector $\dot{\boldsymbol{x}}$. Then the one-by-one-constraints are trivially satisfied. And because we have that $z_s = \dot{x}_{\max} < 1$, for the sum-constraints we get:

$$\sum_{p \in P^{(s)}} \dot{x}_p = \dot{x}_{\max} \cdot |P^{(s)}| > z_s \cdot \dot{x}_{\max} \cdot |P^{(s)}| = z_s \cdot M \tag{3-30}$$

Which concludes the proof. □

From Proposition 3-4.5 it seems reasonable to assume that the sum-constraints are the better choice from the point of view of the solution time. However, as is well-known in the field of optimization, distinguishing between what works well in practice and what does not, is part of the mathematical modeling art and is not an exact science. This is confirmed by analyzing the solution process for a small test case. In this test case, the input data is chosen such that it is representative of larger cases, and the solution time is relatively small. When we apply the two methods, using one-by-one-constraints or sum-constraints, to this case we see that the LP-gap of the model with the sum-constraints is indeed smaller than that of the model with the one-by-one-constraints (resp. 131% and 221%). But after the pre-solve process of GUROBI (see 2-1), these gaps have become resp. 74% and 47%. These gaps then decrease with a similar rate until the optimum is reached. This reduced LP-gap then seems to be the reason for the shorter solution time. To figure out how the pre-solve does this, we can look at the characteristics of the model before and after the pre-solve. They are shown in Table 3-4, where the number of nonzeros is the number of terms in the objective function.

Table 3-4: Effect of the pre-solve procedure on the size of the model. The number of non-zeros, is the sum over all variables, of the amount of times a variable is present in all constraints and the objective function.

	Number of variables		Number of constraints		Number of non-zeros	
	Sum	One-by-one	Sum	One-by-one	Sum	One-by-one
Pre pre-solve	6486	6630	6917	13069	40910	46025
Post pre-solve	4478	4978	5375	10153	26981	33214

These results again seem to be in favor of the sum-constraints, but instead they show that the size of a model says little to nothing about the difficulty of solving it. About the reasons why the pre-solve decreases the LP-gap by more in one case than in another, not a lot can thus be said. As is also explained in Subsection 2-3-2, during the pre-solve an unknown collection of reductions is applied to decrease the size of the problem and tighten its formulation. These routines are heuristic in nature, which is why we can only conclude that in this model they respond better to the one-by-one-constraints than the sum-constraints. Our presumption is that in the case of the one-by-one constraints the pre-solve can combine multiple constraints

to form stronger ones, where in the case of the sum-constraints the options for this are more limited.

Soft capacities An alternative option of implementing the hub capacity constraints is removing the constraints and penalizing the difference between the flow and the capacity. For this approach it is very important to chose an appropriate value for the penalizing factor in the objective function. It should ensure the capacities are not, or minimally, exceeded, while having a good effect on the solution time. Experiments on multiple cases with a wide variety of penalties, have shown that this method does not produce favorable results compared to the use of the normal capacity constraints.

Strengthening In some problems, strengthening the LP-relaxation is a very effective procedure for speeding up the solution process, see for example Gendron (2011). However, in other problems it just makes the model bigger and more complex at each node of the B&B tree. To find out if it is a good option for our problem, attempts have been made to come up with effective valid inequalities, i.e. inequalities that cut off at least one solution from the LP-relaxation while not changing the feasible region of the MILP model, that are not generated by the GUROBI solver. The attempts were mainly based on existing literature on similar problems. Unfortunately no valid inequalities were found that have a positive influence on the solution time. The main reason for this is that all market shares are allowed to be zero, which makes it hard to find inequalities that make solutions to the LP-relaxation unfeasible.

3-4-7 Influence of including the setting of prices

Problems as the one stated in this chapter are regularly solved by ORTEC, with the only exception that in those problems, the setting of prices is not taken into account. The flow between depots is then constant, which makes the revenues constant, and this means that maximizing the total profit means that the total costs are minimized. The main decisions in these problems are the choosing of paths and the opening of arcs. To find out what the difference in solution time is between problems in which the prices are set, and ones where they are not, we have compared them in Table 3-5. These results depend, besides on the actual price-market share relation, also on the price that is used in the problem where the prices are not optimized (because the prices determine the flow). For this, we have taken the prices from the sub-problems of choosing the optimal market share for an isolated commodity if only the revenues and variable costs are included.
The table shows that the differences in solution times are very large, and this means that including the the price setting will be practically impossible in many situations. As an alternative, it is also possible to not determine the exact prices, but to only decide if a commodity should be offered or not. This is covered in Chapter 5.

Table 3-5: The effect of including or excluding the price setting on the time it takes until the optimal solution is found.

	Case a	Case b
Amount of paths per commodity	Small	Large
Hub capacities	Low	High
Solution time with price setting [s]	3920.7	2328.7
Solution time without price setting [s]	7.7	4.5

3-5 Sensitivity analysis

In this section the sensitivity of the solution time of the proposed method (as described in the previous section) to variations in the input data is analyzed. We have tried to ensure that the conditions (for instance background processes that are running on the computer) between different runs are the same, but this cannot be guaranteed and this decreases the reliability of the results slightly.
We will analyze the influence of three main aspects of the model; the first is the hub capacities, the second is the total flow per commodity, and the third is the prices. For the hub capacity we will set all capacities to a deterministic value and then vary this value. For the total flow and the price we will randomly generate data and we will look at variations of the expected value and variations of the variance. For all aspects we will generate data that is based on what is realistic for actual networks. By this we mean that we estimate what is a realistic combination of values for a network of this size, and then vary the values around this point. One example is the data for the price; it is chosen such that the total margin (the total profit as a percentage of the total revenue) is realistic. Another example is the total flow; a company with a high market share in Europe would have a high expected total flow and low variance of the total flow for depot-depot combinations (because Europe is densely populated all depots serve a similar amount of customers). On the other hand, a company in the U.S. with low market share would have a low expected total flow and high variance since the difference between the number of assigned customers per depot is greater than in Europe. For every case we will generate and analyze multiple generated data sets to decrease the probability that an outlier is taken as a general case.

The ranges of values for which the performance is analyzed is covered in the corresponding paragraphs. A general approach is that for parts of the ranges that show results that diverge from the general trend, more points are analyzed than for other parts. This results in a non-uniform step size when varying the input data, but enables us to draw stronger conclusions from the graphs because they give more detailed information on the most important parts.

The approach formulated in the last section uses, next to the solution process of the MILP model, the solution process of the LP relaxation and a procedure to pre-process the set of potential arcs. These parts of the approach are ignored in the computational experiments

since for larger networks the duration of these processes is negligible, and for any variation in input data they are roughly the same.

Other settings To be able to analyze the performance for many different cases, the input data other than the three parts of this section and the settings of the model are chosen such that the solution time is relatively small, but the resulting model is representative for bigger ones. Firstly, the pool of pre-selected paths is small, for every commodity there are three potential paths. Secondly, the non-linear objective function is linearized by using only four lines. And thirdly, the arcs are pre-selected from the solution to the LP-model as explained in Subsection 3-4-3. Furthermore, running the model is done without tuning the solver as described in the previous section. We are interested in the fundamental causes behind the computational performance and this is best analyzed when the same methods and setting are used for every case.

3-5-1 Hub capacities

A big influence on the solution time is the setting of the capacities of the hubs. When the capacities are large enough so that they are not reached in the optimal solution, the solution time is many factors smaller than when they *are* reached. When they are that large the market share for all commodities can be at their local maximum, and the chosen path can be the lowest cost path (provided the right arcs are open). When the capacities are lowered the model has to make more and more decisions on what market shares to decrease or which alternative paths to choose. The main driver for increases in solution time are the amount of these decisions that need to be made, the amount of options there are for the decisions, and the difficulty of choosing between them (how "close" are the options to each other with respect to the objective function).

The data that is used for the other parts that are covered in this section, the total flow and the price, is generated with low variance. The capacities of all hubs are set at the same level. The results are displayed in Figure 3-12, where the results are shown for different price and total flow levels. Values for which the solution time is larger than 12,000 seconds were not analyzed. What we see, is that in general the tighter the hub capacity constraints are, the longer it takes to find the optimal solution, and that the solution time sharply increases at a certain point when lowering the capacities. Also, the higher the prices and total flow, the "lower" the point is located where the solution time sharply increases. The location of this point depends on the price and total flow factor. When these factors are 0.8, and we start lowering the hub capacities, this point is reached when *half* of the hubs have a flow that is equal to their capacity, and that increases up to the point when these factors are 1, and *all* of the hubs have a flow that is equal to their capacity. This is not what we initially expected. We expected this point to "move right" in the graph if the prices and total flow are increased, since higher total flow and higher prices mean that there is more flow in the network and thus that the capacities play a more important role. The explanation that the point "moves left"

is in Subsections 3-5-2 and 3-5-3, where we will see that if the hubs do not have a capacity, as the prices and total flow increase, the solution time decreases. We can conclude that this last phenomenon influences the solution time more than the fact that more total flow result in a larger solution time.

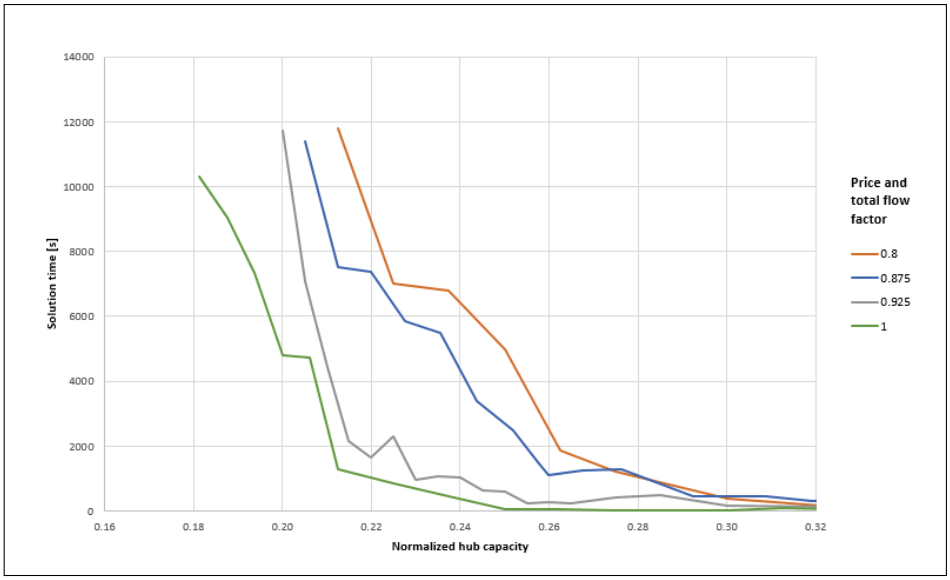


Figure 3-12: Solution time in relation to the hub capacities. The price and flow factors represent the relative difference between the prices and the total flow of the different cases.

3-5-2 Total flow

When we are analyzing the influence of the level of total flow, we vary the total flow that is sent from all depots combined, and we do not change the relative distribution between the depots. The results of this subsection also say something about the influence of the price-market share relation on the solution time. This is because the total flow of a commodity is multiplied by the market share in the model, which is why a large total flow means the average gradient of the price-market share relation is steep and vice versa (as a reminder: the relation is non-increasing and concave). Similarly, if the variance of the total flow between the commodities is large, the results also hold for large variance in the average gradient of the price-market share relation.

We have seen in the previous subsection that the influence of the hub capacity setting is very large. This fact will also play a large role in analyzing the performance as a function of variations in the total flow. If the hub capacities are at some fixed level and the total flow is increased, the actual transported flow will also increase. Then the solution time will increase, but this will then mostly be due to the capacity constraints that become increasingly tight. However, since this effect has been studied in the previous subsection it should not be the focus of this subsection. Here, we would like to analyze the performance in response to the

interplay between the total flow and the rest of the model. This is why we will solve the model without capacity constraints on the hubs. The prices are such that they give average results in the computational experiments.

The flow data is generated in a way that ensures it resembles real data. First, all depots are assigned a random value from a uniform distribution in a range that is dependent on the variance setting. This value represents the "size" of the depot, so how much flow will go through it. A large value could for instance mean it is located in a big city. For the smallest variance this range is $[0.9, 1.1]$, and for the largest variance this is $[0.2, 1.8]$. This value determines a total flow that is sent from a depot. Then the flow to each destination depot is determined by the fraction of the random value of the destination depot with respect to the total of all random values. This way a large depot will receive a large portion of the flow that is sent from the origin depot, and a small depot will receive a small portion.
The results are shown in Figure 3-13-a.).

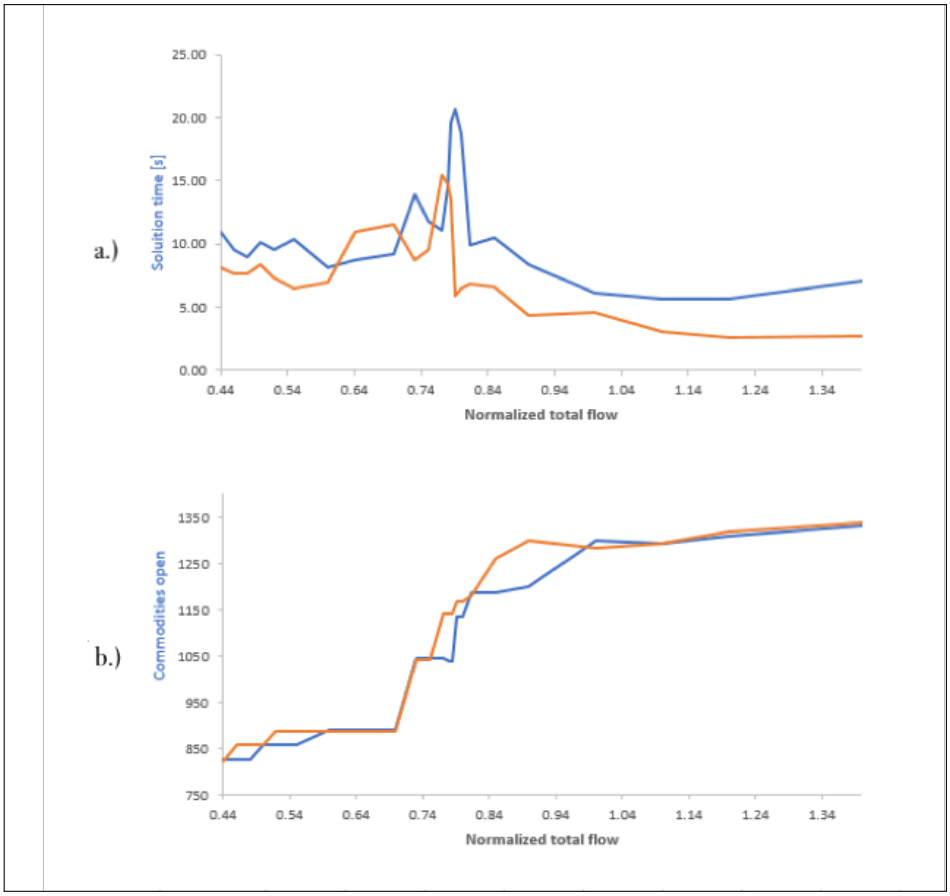


Figure 3-13: a.) Sensitivity of the solution time to variations in total flow.
b.) Number of commodities with positive market share in the optimal solution as a function of the total flow.

We see that in general the amount of total flow does not have a big influence. In a small

range around 0.75 the solution time is a little unpredictable, here the solver has a harder time deciding which commodities to open. This is illustrated by Figure 3-13-b. which shows that within that range, a lot of commodities change from being unprofitable to being profitable.

When we do this experiment for different values of the price, we get similar results, as is shown in Figure 3-14-a. We see that in general it holds that the higher the price, the lower the solution time. This is likely caused by the LP-gap, which shows a similar relation (see Figure 3-14-b.). From the LP-gap one might expect the solution time to be large for small values of the total flow, but this is not the case because for these values the solver quickly concludes that the solution where all market shares are zero is optimal. We also conclude from Figure 3-14-a. that the difference in solution time between the best and worst cases is small. This conclusion only holds for the situation where the hub capacities do not play a big role, otherwise the difference between the best and worst cases will likely be a lot bigger.

Next, we look at the influence of the variance of the total flow on the solution time. The results are shown in Figure 3-15. For this part the hub capacities *are* taken into account, the total flow does not change between the different cases and so the capacities are expected to have a similar influence on every case.

More variance in the total flow results in a more unpredictable solution time. This is because a high variance makes it possible that the flow is spread over the network in a way that makes dividing the flow over the hubs either easy or hard. We also see that the relative influence on the capacitated case is larger than on the uncapacitated case.

3-5-3 Prices

By varying the prices, we vary the amount of revenue that is earned per unit of weight of parcels. And for the influence of variations in the price on the performance a similar statement holds as for the influence of the total flow. Namely that the price influences the flow, and this in turn determines a large part of the solution time because the hub capacity constraints become more or less tight. To deal with this we use the same approach as with the total flow, namely run the model without capacity constraints. The other part of this section, the total possible flow, is again chosen such that it gives average results.

Again the prices are generated in a way that resembles real data. First, for every depot a random value is uniformly generated from a range that varies between $[0.75, 1.25]$ and $[0, 2]$. The average value of an origin-depot and a destination-depot gives us a factor that determines the price that is asked for sending parcels between these depots. This factor is then multiplied by a base price which is dependent on the distance between the two depots and the service (e.g. a one-, two- or three-day service). The expected value of the price is varied by varying the base price.

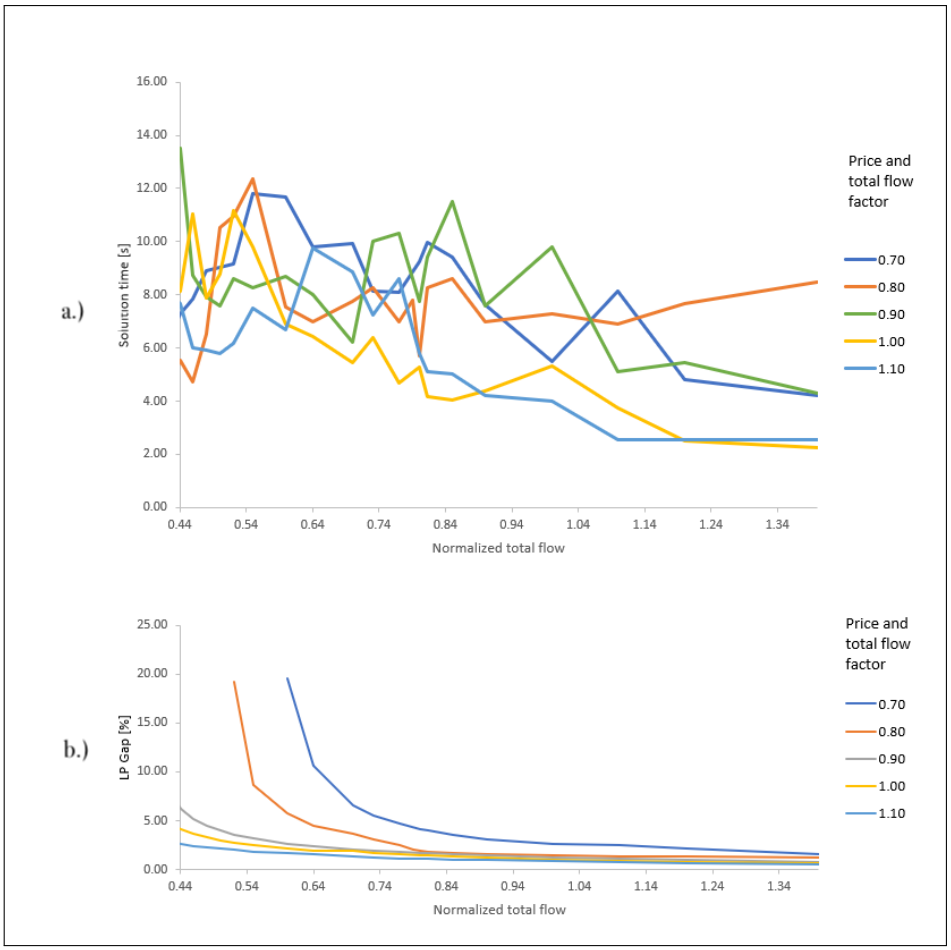


Figure 3-14: a.) Sensitivity of the solution time to variations in total flow for different price settings.
b.) LP-gap as a function of the total flow for different price settings.
The price and flow factors represent the relative difference between the prices and the total flow of the different cases.

Based on these experiments we can also say something about the influence of variations in the variable costs. There are two parts of the model that are directly dependent on the amount of flow that is transported, those are the variable costs and the prices. In the objective function there is a product of the flow with both these parts for every commodity. That is why we can say that if prices are increased and the variable costs stay the same, the variable costs get relatively smaller. So the conclusions drawn from testing the performance of the model under increasing prices are also applicable to lowering the variable costs.

First we discuss Figure 3-16-a, where the prices of all commodities are varied in unison. We see that as prices get lower, it is increasingly difficult to get to the optimal solution. This makes sense as the decisions are increasingly difficult if the commodities have a price-cost ratio that makes it harder to decide if a commodity is profitable or not. We also see that

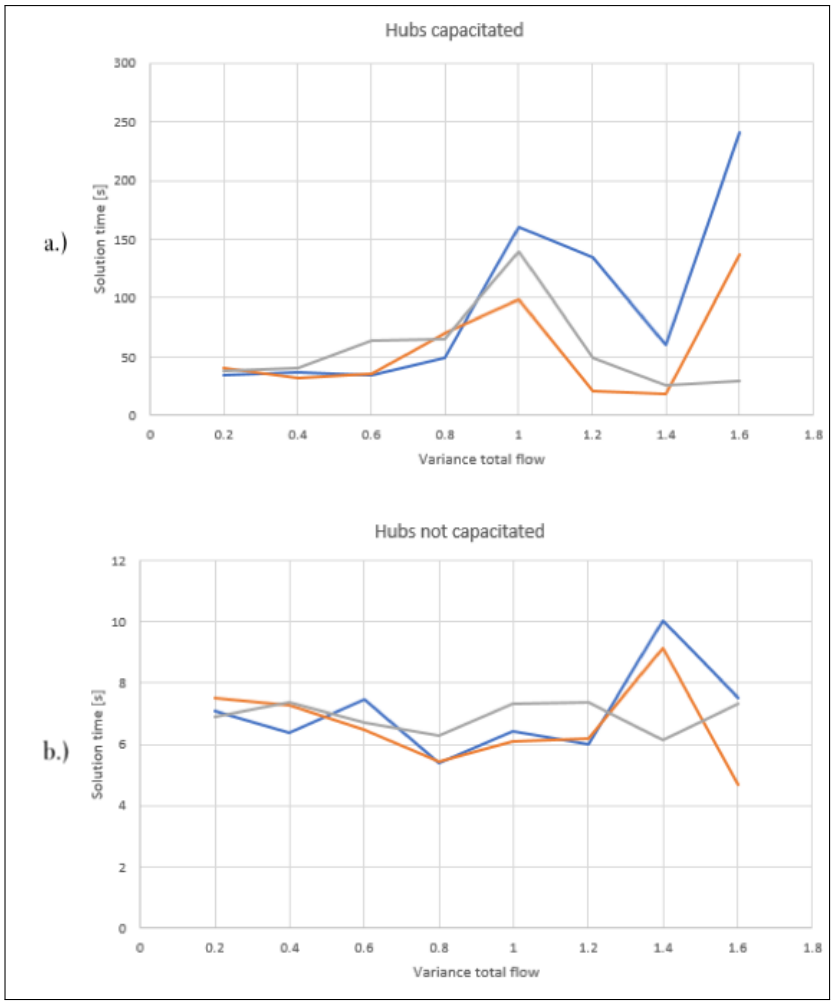


Figure 3-15: Sensitivity of the solution time to variations in the variance of the total flow for three different input sets for:
a.) The situation where the hubs have a capacity,
b.) The situation where the hubs do not have a capacity.

the LP-gap (Figure 3-16-b) increases as prices get lower. As the prices only influence the objective function of the model this must mean the optimal solution is in a more difficult part of the feasible region. The reason that for very low prices the solution time is low, is because at a certain point the optimal solution is the one where all market share are zero, which is illustrated in Figure 3-16-c.
The consequences of different settings for the variance of the prices in the generated input data are given in Figure 3-17. These show comparable results to the cases where the variance of the total flow is varied, namely a higher variance leads to a more unpredictable solution time.

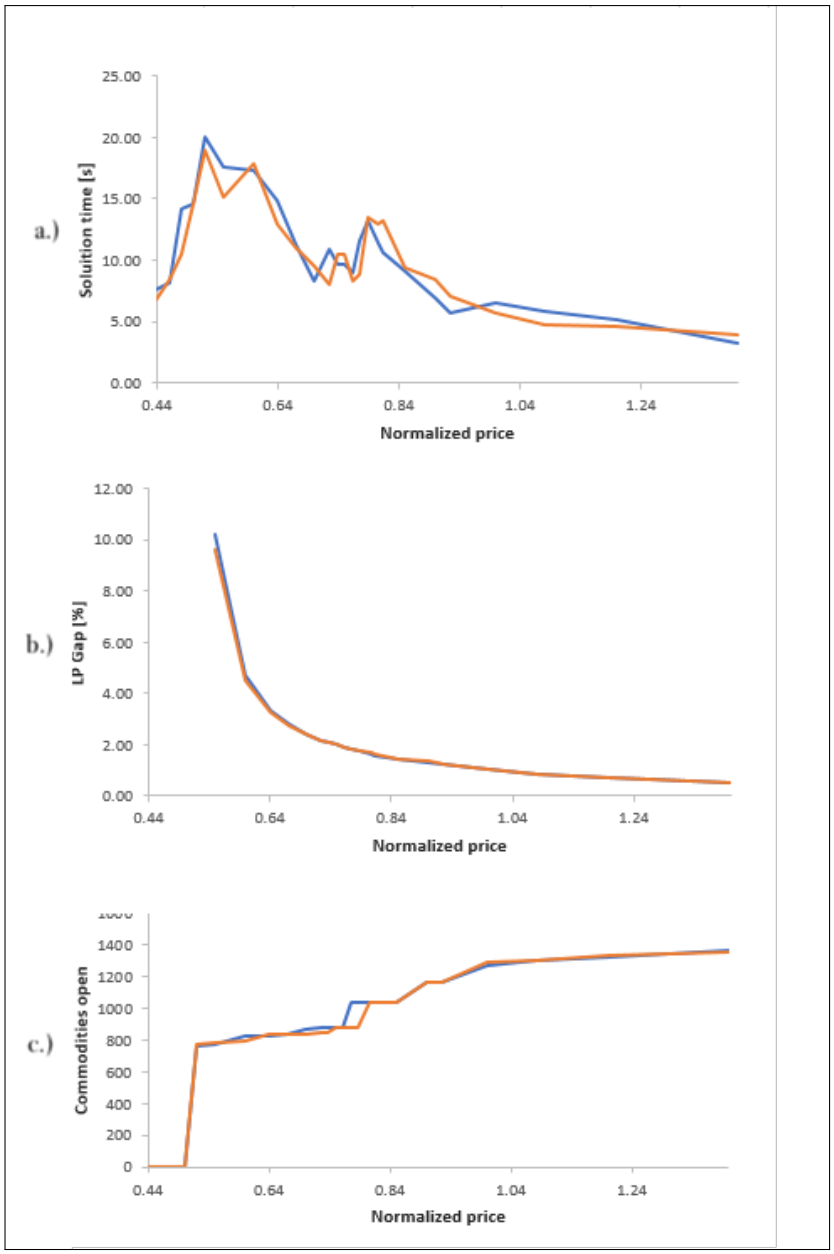


Figure 3-16: Analysis of the solution time in response to variations in the price.
a.) Solution time as a function of the price.
b.) LP-gap as a function of the price.
c.) Number of commodities with positive market share in the optimal solution as a function of the price.

3-5-4 Conclusions

The most important conclusion is that the setting for the hub capacities is the most significant contributor to the solution time. The values that are chosen for this, in relation to the amount of flow in the network and the prices, determine for a large part how long it takes to find

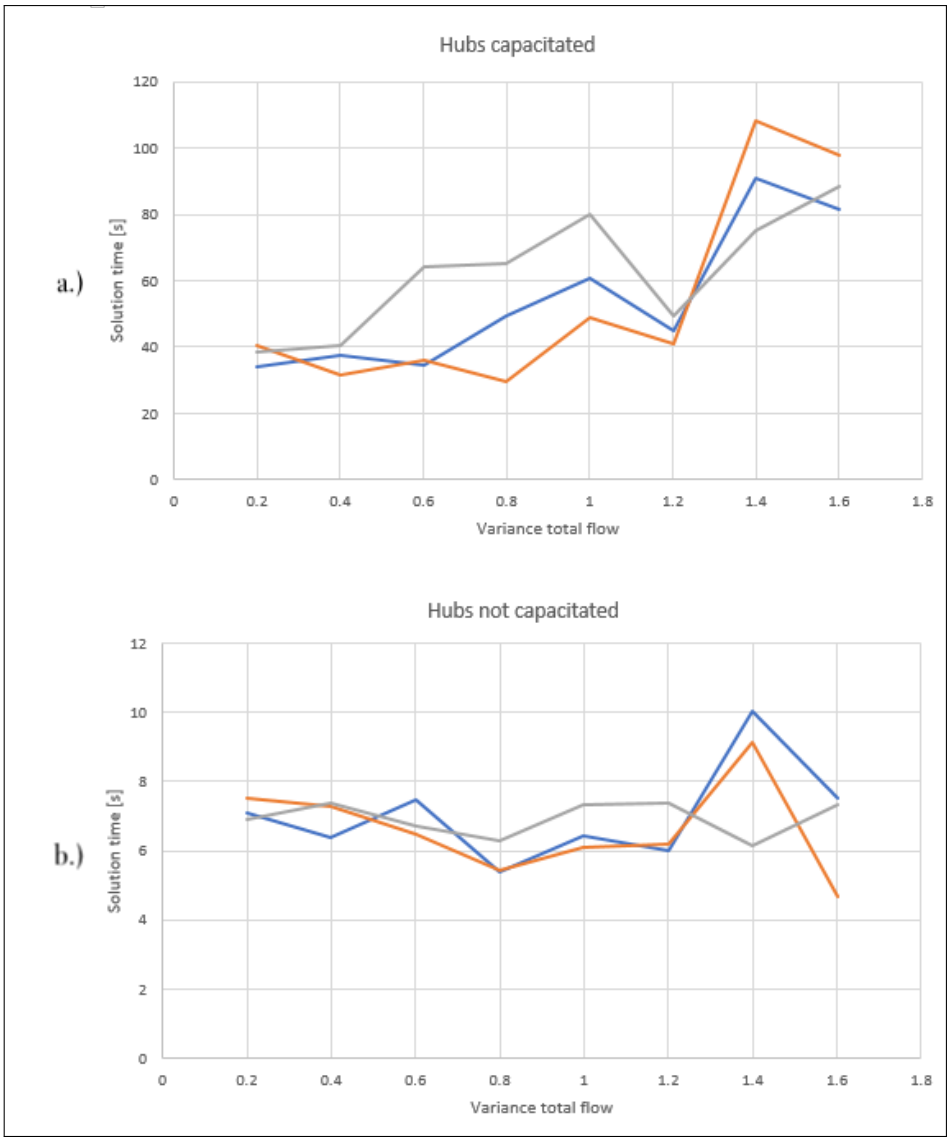


Figure 3-17: Sensitivity of the solution time to variations in variance of the price for:
a.) The situation where the hubs have a capacity,
b.) The situation where the hubs do not have a capacity.

the optimal solution. The prices and total flow in themselves have a small influence on the solution time in an absolute sense, although the difference between the lowest solution time and the largest solution time is approximately a factor of 5. However, this becomes relevant if the solution time is larger due to hub capacities (or a larger network). Then the price and total flow determine at what capacities the solution time becomes very large.

Chapter 4

The Variable Network

4-1 Introduction to the variable network

In this chapter, the second situation of the research question is covered. We are again looking for a solution approach for finding the optimal pricing strategy and network structure. Compared to the previous chapter, the network has less parts that are fixed, which gives more options for optimization. The extra binary variables that are involved increase the complexity of the model significantly. The situation is as follows: The number of hubs and their locations must be chosen, and for every hub there is a choice between three different sizes that each have a different capacity. Furthermore, depots and arcs can be opened or closed. An overview of this is shown in Table 4-1.

Table 4-1: Implementation of variables in the Variable Network.

Variable	Range	Domain	Comment
Market share	Continuous and bounded	Set of potential paths	Also acts as path-choice variable
Arcs	Binary	Set of potential arcs	-
Depots	Binary	Set of potential depots	-
Hubs	Binary	Set of potential hubs	3 possible sizes

The rest of the chapter is arranged as follows: In the first section, a mathematical formulation of the problem is given. In the second section, a review of related literature is covered, along with the contributions of this chapter and some challenges as presented in Chapter 3. This is followed by the solution method in the third section, and the application of the method to a test problem in the last section.

4-2 Mathematical model

In Chapter 3, the non-linear model was approximated by a linear formulation, and a method was developed for finding a solution to this MILP model. As the problems are very similar, the same approach will be used in this chapter. For the details on how the non-linear model is transformed to a MILP model, see Section 3-4.

A MILP formulation The formulation of the MILP model is very similar to formulation (3-12) to (3-20). To easily see the differences between them, we have only shown the modifications that have been done to the first formulation:

- In the objective function, terms are added for the depots:

$$+ \sum_{d \in D} z_d c_d,$$

  where $z_d \in \{0,1\} \ \forall d \in D$, and c_d is the cost of opening depot d.
- Constraints are added that ensure that over a path p, there can only be positive market share if the origin and destination depots or p are open:

$$\dot{x}_p \leq z_d \cdot \dot{x}_{max}(p) \qquad \forall d \in D, p \in P^{O(d)} \cup P^{D(d)}$$

  The multiplication with the maximum market share in this constraint strengthens the formulation.
- In the objective function, terms are also added for the hubs:

$$+ \sum_{h \in H, \bar{h} \in \bar{H}} z_{h\bar{h}} c_{h\bar{h}}$$

  where H is the set of potential hub locations, $\bar{H}$ is the set of potential hub sizes, $z_{h\bar{h}} \in \{0,1\} \ \forall h \in H, \bar{h} \in \bar{H}$, and $c_{h\bar{h}}$ is the cost of opening a hub at location h with size $\bar{h}$.
- Constraints are added that ensure at most one hub size is chosen per location:

$$\sum_{\bar{h} \in \bar{H}} z_{h\bar{h}} \leq 1 \qquad \forall h \in H.$$

- And finally, Constraints (3-7), which model the capacity of the hubs, are modified into the form:

$$\sum_{k \in K} \sum_{p \in P^{(k)} \cap P^{(h)}} f_k^{pcs} \cdot m(x_k) \cdot y_{kp} \leq \sum_{\bar{h} \in \bar{H}} z_{h\bar{h}} C_{h\bar{h}} \qquad \forall h \in H,$$

  where $C_{h\bar{h}}$ represents the capacity of the hub at location h with size $\bar{h}$.

Modeling choices As the model is very similar to the model in the previous chapter, the choices regarding the modeling of the business problem are the same, and they are covered in Subsection 3-2-2.

4-3 Literature

Here, an extension of the literature review in Section 1-4 is given that is specific to the second situation as mentioned in the research question.

4-3-1 Context

A major topic in network design is the decision on the location of hubs (both single and multiple) in a network, and it is also an important part of the current research question of this thesis project. It has been studied extensively, and Campbell and O'Kelly (2012) give a comprehensive overview of twenty-five years of research into the subject. The combination of hub allocation and price setting has had significantly less attention. Here, competition between companies is an important part of the problem. This has mainly been studied from a sequential location approach (Luer-Villagra and Marianov (2013)), where an existing firm, called the *leader*, serves the demand in a region, and a new firm, or *follower*, wishes to enter the market. In the literature this is called a Stackelberg competition. An example is the study by Gelareha et al. (2010), where the competition between a newcomer maritime liner service provider and an existing dominating operator is studied. Here, the follower aims at designing a network that maximizes the total profit, and market share is assumed to depend on the service time and price. A MILP model is introduced and a Lagrangian-based decomposition method is used for obtaining good lower bounds.
A study by Luer-Villagra and Marianov (2013) also simultaneously considered hub location and pricing. The case is that of an airline follower company using an incomplete HS-network. Customers choose a company and path solely based on price, which is modeled through a logit function. The researchers propose a non-linear model and use an ad hoc metaheuristic that at each step finds feasible solutions for the hub location problem and, for each such solution, solves a pricing problem.
Mahmutogullari and Kara (2015) deal with two problems in their research into hub location when competition is present. The first assumes that the leader in the market has already selected his hub locations, which gives the follower information about the service levels of the leader, and then optimizes the locations of the hubs of the follower. The second problem is from the perspective of the leader and optimizes the second choice for the locations of the hubs, knowing that the follower will use the results of the first problem that was just mentioned. In both cases the customers choose a company based on the price.

4-3-2 Challenges

Solution methods For hub location problems, which is an important part of the this Chapter's research question, Benders Decomposition is a much used solution technique in the literature. It is a partitioning method first proposed by Benders (1962), and for appropriate problems good, or even the optimal, solutions can be found in relatively few iterations. It is for example used by Contreras et al. (2011), where the technique is further enhanced and an exact algorithm is given, which is applicable to large-scale symmetric instances involving

up to 500 nodes and 250,000 commodities. The reason that the technique is successfully applied to these problems is because of their structure, which can be seen to have two levels of decision-making. The first level determines the location of the hubs, whereas the second level deals with the optimization of the hub network, which in most studies is defined as the allocation pattern of nodes to hubs. The classical variation of the method separates the problem into two simpler ones by partitioning the variables into two sets, the first gives an integer *master-problem*, the second a *sub-problem* that is either a MILP problem or a LP-problem. First, the master-problem is solved to optimality, and with these variables set, a bound on the optimal objective value is found by solving the dual of the sub-problem. Using the values found for the sub-problem-variables, a so called *Benders cut* is generated, which is then added to the master-problem.

4-4 Solution approach

Based on the previous chapter we can safely assume that solving the MILP model to optimality will not be possible in any realistic situation. As discussed before, an option is stopping the solution process before the proven optimal solution is found. However, although a general-purpose MILP solver can be configured to find good quality solutions early in the process, it is designed to solve a very broad spectrum of problems. This leads us to believe that it is possible to design an algorithm that finds better quality solutions quicker. That is why we will use problem-specific knowledge and concepts from the field of meta-heuristics, and in particular local search, to try to design an algorithm that performs better than the GUROBI-solver in the sense of quickly finding good quality feasible solutions. We refer to Section 2-2 for an overview of the concepts of meta-heuristics, and local search and some of their variations.

4-4-1 Framework

Solving the model can obviously be done very quickly when all binary variables are fixed, since the resulting model is an LP-model. Therefore, we will design an algorithm that finds good values for these binary variables quickly. These variables consist of the hub, depot, and arc decisions. We note that the model naturally decomposes into multiple levels; On the higher levels there are few decision variables, and they have a strong impact on the objective function. On the lower levels there are a lot more variables, and they have a decreasing amount of impact on the objective function. The decision variables ordered from the high level to the low level are: Hubs-depots-arcs.

We have chosen to base the algorithm on VNS (see Section 2-2), but elements from other algorithms such as TABU search and LNS are also used to create a hybrid meta-heuristic algorithm. A typical meta-heuristic algorithm that uses local search consists of two phases; the phase where *starting solutions* are generated, which is called the diversification phase, and the phase where the neighborhood of those solutions is searched for a better solution that

then becomes the new incumbent solution, which is called the intensification phase. Since the model decomposes into multiple levels, it is a reasonable approach to use the higher levels to define starting solutions, and the lower levels to find the local optima. This means we will find starting solutions that have as their main characteristic the hub locations and sizes, then find the best set of depots to open, and then use a local search procedure that mainly finds better sets of arcs to open. It can also change the hubs and depots, this will be further explained in the next subsection.

To define the neighborhoods in the MILP model, a so-called *local branching constraint* is introduced. This idea was originally introduced by Fischetti and Lodi (2003). The constraint defines the kth neighborhood "around" a solution by requiring that the amount of binary variables that are different in a new solution is less than or equal to k. This difference is called the *Hamming distance* and is represented by $\Delta(x, y) = \sum_{i \in \mathscr{B}} |x_i - \tilde{x}_i|$, where $\mathscr{B}$ equals the set of all binary variables. Given the MILP with variables x and a solution $\tilde{x}$, the constraint is linearly defined as

$$\Delta(x, \tilde{x}) = \sum_{\tilde{i} \in \mathscr{S}} (1 - x_i) + \sum_{\tilde{i} \in \mathscr{B} \setminus \mathscr{S}} x_i \leq k$$

Where $\mathscr{S} = \{i \in \mathscr{B} | \tilde{x}_i = 1\}$. This approach allows the use of a variety of local search based heuristics such as Tabu Search, Simulated Annealing, and VNS (see Section 2-2).

In a similar way, the solutions in a neighborhood can be excluded from the feasible region. We call these constraints TABU-constraints, and these are formulated as

$$\Delta(x, \tilde{x}) \leq k$$

Note 4-4.1. A characteristic that we would like the algorithm to have, is that as many parameters as possible should be automatically chosen based on the problem that is fed to the algorithm. This is because the performance of heuristic algorithms is heavily dependent on the chosen parameters, and parameters that work well for one problem might result in very bad performance for another problem. Manually discovering the right parameters is impossible when the run-time is long and the optimal solution is not known, so ideally the algorithm finds the right parameters itself.

4-4-2 The algorithm

In Figure 4-1, an overview of the algorithm is given. As the total amount of time that the algorithm is allowed to take will be limited in most cases, a balance needs to be found between the time spent on the diversification and intensification phase. To accomplish this, the algorithm starts by generating m starting solutions, after which the intensification phase commences. Depending on the total time limit and whether it is reached when all hub configurations reach stage 4, additional starting solutions may be generated later in the algorithm. The value of m is dependent on the specific network structure and the number of potential hubs. A high

value for m means a large amount of time is spent on finding new hub configurations, which leaves a relatively small amount of time for the intensification phase. However, if the value for m is high, we also see that the probability is high that a solution is found that has a hub configuration that will lead to a good quality solution after the intensification phase. The choice of m is a trade-off between this probability and the relative amount of time that is spent on the two phases phase.

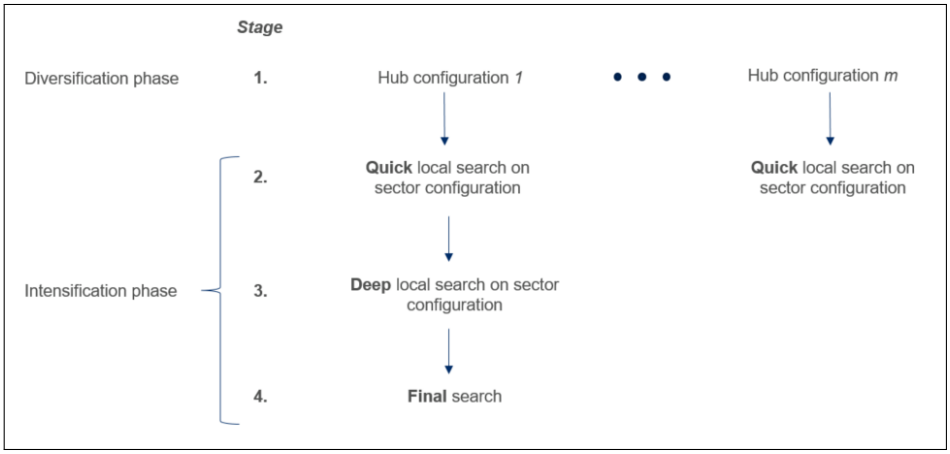


Figure 4-1: An overview of the algorithm. Stages 3 and 4 are first performed on the "best" hub configuration, then on the second "best", and so forth.

We want to mention here, that in realistic situations, there will be a pool of hub configurations that have a very similar cost structure. This phenomenon is known in literature as the "batchtub-curve" (see Figure 4-2 and for instance Meuffels (2010)). For our algorithm, this means that, to be able to find a good final solution, we aim at including at least one of the good configurations from the bathtub in the m solutions.
The intensification phase (or getting from a starting solution to a final solution) requires a large amount of time. Considering the limited amount of time that is available for the algorithm, we would first like to use the intensification phase on the "most promising" solution among the m starting solutions. To find this solution, a *quick* local search is done (stage 2) on all m solutions, which finds quick improvements to the solution. After this has been done, the solution with the highest objective value is chosen as the solution to perform stage 3 on first.

Stages Below, the different stages, as they are depicted in Figure 4-1, are described.

Stage 1
At the first stage, feasible solutions (which we call starting solutions) to the MILP model are found that are characterized by their hub configurations. We define a hub configuration as the location and size of all hubs. To find these solutions, a solution process is performed with settings that ensure a solution is found that is "somewhat close" to the optimal solution, which thus has a hub configuration that is probably close to the one in the optimal configuration. To determine when to stop the solver, and to accept the best found solution as the

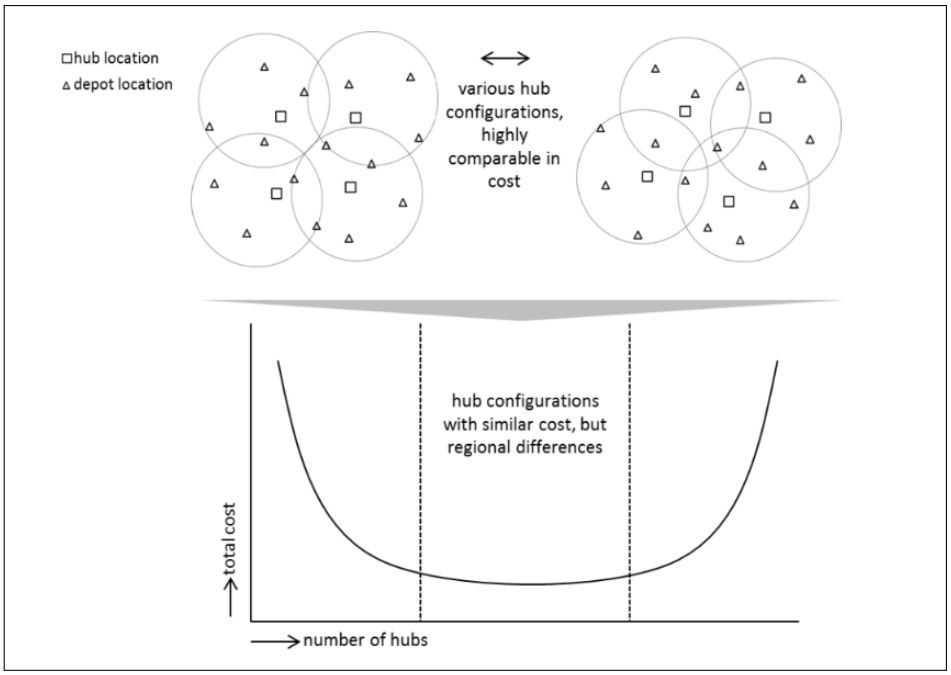


Figure 4-2: The "batchtub-curve": the hub location and network design of express service providers results in a range of solutions that are comparable in cost, and show regional differences only (Meuffels (2010)).

starting solution for the next stage, a stopping parameter is used that is discussed later in this section, such that only the biggest improvements at the start of the solution process are performed.

In the solution that is found in this stage, a subset of the depots is open. To improve the performance of this solution process, without negatively influencing the solution that is found, we can help the solver in the right direction with some problem specific knowledge. We know that in the final optimal solution most of the depots will be open. So a constraint is added that ensures the most-potentially-profitable depots are open. This is based on the costs of the depot itself, the adjacent arcs, and the commodities that have the depot as their origin or destination. This way, the MILP model contains less variables and is thus faster to solve.

Furthermore, for finding hub configurations when at least one configuration has already been found, two additional constraints are added. The first one excludes solutions with previously found hub configurations. Besides this, not only these solutions are excluded, solutions that have one or more hubs at the same location but with a different size, are also excluded. This is done because of the "batchtub-curve", and the fact that, as we will see later in this section, changing only the size of a hub can be done in the local search phase, which means a solution with a hub configuration where only the sizes of hubs are different does not have to be in a separate starting solution. The second constraint that is added here is a local branching constraint, with a large k, around the best solution that was found thus far. Good solutions with new hub configurations lie relatively close to other good solutions, this constraint ensures the feasible region is limited to the relevant solutions.

Stage 2
In this stage, the goal is to improve the solution that was found in the previous stage, which is now the current solution, as much as possible and in a short amount of time. In a sense, this means we quickly try to "move" to the region of the solution space where the best solutions lie. For this, the local search procedure is used that is described in the box below. The time limit for a single solution process in this stage is small.

Local search In stages 2 and 3, a local search procedure is used. This consists of a series of solution processes, that iteratively improve the *current solution*. The solution processes are done on the original MILP model with an additional local branching constraint, around the current solution, with parameter k. The changes in the solution that are allowed are opening and closing of depots and arcs, and changing the size of the hubs. A time limit t is set for how long a solution process can maximally take in this stage. The choice of t ensures that at least one solution is found during the solution process. And so, for a solution process, there are four possible results, and each has an associated next step (e.i. $\rightarrow$):

1. The local optimal solution is found before t is reached, and it is better than the current solution.
   $\rightarrow$ A TABU-constaint is introduced around the current solution with parameter $k-1$, and the found solution is accepted as the new current solution.
2. The local optimal solution is found before t is reached, and it is worse than the current solution.
   $\rightarrow$ A TABU-constaint is introduced around the current solution with parameter $k-1$, and the k is increased.
3. Time t is reached and the found solution is better than the current solution.
   $\rightarrow$ A TABU-constaint is introduced around the current solution with parameter 1, the found solution is accepted as the new current solution, and k is set at its initial value.
4. Time t is reached and the found solution is worse than the current solution.
   $\rightarrow$ The local search is terminated.

The TABU-constraints exclude solutions that are guaranteed to be worse than the current solution from the MILP model. The increase in k ensures solutions that are further away can be found, if all solutions that are close are guaranteed to be worse than the current solution. If no solution can be found that is better than the current solution we assume the current solution is the local optimum.
Time limit t is an upper bound on the time that a solution process is allowed to take and determines how "thorough" the search for the local optimum is for a specific neighborhood. For every solution process, we want to set it to the appropriate value automatically. This is why it depends on the solution time of the preceding iteration in the local search.

If the time that is needed to find a better solution than the current solution is a lot longer than the average of the preceding iterations, we assume no better solution will be found, and the solver is stopped. This is a reasonable assumption in almost all cases because the average time of the preceding iterations says something about how long it takes until the big improvements are made within a solution process (this is a consequence of the stopping parameter that will be discussed later). In Figure 3-9 we see that, in this representative case, these big improvements are made early in the solution process. This is a characteristic of modern MILP solvers and can thus also be used here. So when we let the solver run for, for instance, 5 times the average time of the preceding iterations, we can assume we are very close to the optimum (which in this case is the local optimum). This is also apparent from Figure 4-3, where we see the solution time and the LS time limit of each iteration of the local search. We want mention that in iteration 15, the optimum was found, and no improvement was made to the current solution. This means that if we stop the solver at the time limit t (where the two lines intersect), the final solution is the same, but the total time that is spent, is significantly shorter.

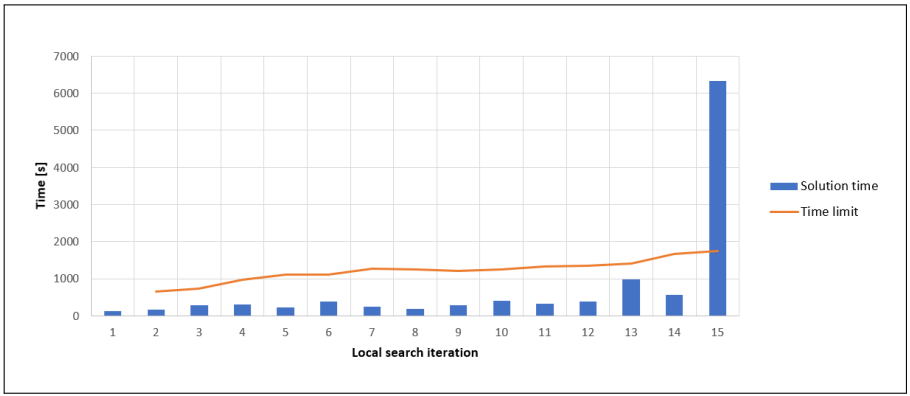


Figure 4-3: The unconstrained solution time and the time limit of the solution processes during the local search procedure.

The "direction" that the local search moves through the solution space is, generally speaking, the direction of *steep* descent. By this we mean that we wait until the solver finds a good improvement of the current solution, but we do not wait until the optimum is found (which would be the direction of *steepest* descent).

Stage 3

A *deep* local search procedure is implemented in this stage. This is the same local search procedure as was discussed in the box above, only it uses different settings. These settings ensure that the procedure continues its search for improvements until we can be fairly certain we are close to the optimum (considering the current hub configuration). The main setting

that ensures this is the stopping time of the solver, which is set to a large value. To increase the "speed" of the algorithm, the first part of this stage is done using only a subset of all arcs as potential arcs. In this subset, we want to include the arcs which are most likely to improve the current solution. The subset is determined by first solving the original problem, with extra constraints such that all hubs and depots from the current solution are fixed, and the arc variables are relaxed. This turns the MILP model into an LP model. One additional constraint is then added, which ensures that all arcs that are open in the current solution, are closed in any LP solution. This way, the arcs that get a non-zero value in the optimal solution, are the arcs that might improve the current solution. The subset of potential arcs is made up of these arcs and the arcs that are open in the current solution. If no better solution can be found in this way, the local search is continued using the full set of potential arcs.

Stage 4
During the last iterations of the deep local search (stage 3), the current solution is already close to the local optimum, which means the solver needs a lot of time to find a better solution (and in difficult problems, the optimum is not found before the stopping criterion is reached). Furthermore, no depot or hub variables and only a few arcs variables are changed during these iterations. This is why it is beneficial to do a separate *final* solution process on the original MILP model, with a smaller value for k, and constraints that fix the depot and hub variables. This MILP model is much easier to solve than the last iterations stage 3, the solver can now be run until the optimum is found.

MILP solution processes In various parts throughout the algorithm, a solution process is performed on the MILP model with specific additional constraints. All these solution processes are done using the GUROBI-solver with settings that are optimized for that specific part. An important example of such a setting, is when to stop the solver and accept the best solution that was found up until that moment. For this, we use one or more lower bounds and an upper bound, which are set automatically and depend on the part of the algorithm in which the process is performed. Something on the upper bound during stages 2 and 3 is covered in the box above.
A lower bound that is used in every solution process is the *stopping parameter*, which is based on the following observation. During a solution process, incumbent solutions are found of increasing quality. The general trend is that the amount by which the objective function of a solution is increased (compared to the objective function of the previous solution) decreases. And that for these improvements, an increasing amount of time is needed. We have captured these two characteristics in a stopping parameter that is high if a large improvement is made in a short amount of time, and vice versa. This parameter is used to prevent the solver from stopping too early (i.e. when quick and good improvements can still be made). For different parts of the algorithm, the best moment that the solver is allowed to stop is at a different time, because the difficulty of the solution processes is different (and thus the improvement-time ratio is different).

4-4-3 Settings

As mentioned in Note 4-4.1 above, we have tried to set most settings in the algorithm automatically. The approach was to find generally applicable representations of settings such that they can be the same for every model that is fed to the algorithm, and then configure them, such that they work best for the goal of the stage that they are used in. For stages 1 and 2 this goal is to quickly move to the region of the solution space where the best solutions lie. And for stages 3 and 4, this is to find a good solution.
One example of this is the stopping parameter as it was previously discussed. Manually deciding when the solution process is allowed to be stopped would require setting a time limit, and this limit would be different for every model. This is generalized by calculating the average improvement over the last two improvements, in percent, per second. If this value drops below a certain threshold the solution process is allowed to be stopped. The best value for this threshold in a certain stage is similar for every model, and the best value was chosen based on test cases.
Another example is the time limit $t = M \cdot t_{avg}$ as it was discussed in the box above. Here M is a factor that is set by finding the best value for several test cases, and t_{avg} is the average solution time of the previous solution processes. A third example is the setting for the size of the neighborhoods in stages 1, 2, and 3. The value that works best for this, is dependent on the network for which the model is solved. The generally applicable representation of this setting uses the size of the network. If there are a lot of arcs, depots, and hubs, the neighborhoods should be relatively large, and vice versa. Again, extensive testing was used to determine the best relation between the size of the network and neighborhood size. Here we took into account that, because the whole neighborhood is searched for good solutions, the larger the neighborhood is, the less likely it is that the local search gets stuck in a local valley.

Two high level settings are not set automatically in the algorithm:

- The total time limit, and total number of hub configurations for which all stages are performed: the algorithm stops when one of the two is reached.
- The number of hub configurations m that are found before one configuration is chosen to perform stages 3 and 4 on. If m is low, and after stages 3 and 4 are performed on a hub configuration i, new hub configurations are found, these new configurations are close to i because of the local branching constraint, and i is likely close to the optimal solution.

4-5 Computational experiments

In this section, we discuss how the algorithm performs. We will compare it to the GUROBI solver in terms of the quality of the solutions that are found during the execution, and the time that is needed to find them. Both the solver and the algorithm are tested on a relatively small network, and cases with varying complexity are generated which are shown in Table

Table 4-2: Definition of the cases in regard to testing the algorithm.

	Case a	Case b
Number of paths per commodity	7	5
Number of lines in the linearization	5	10
Factor for total flow	1	1.2
Normalized hub capacities	1	1.08

Settings The following high-level settings are used for the algorithm:

- The total number of hub configurations for which all stages are performed is set to 2. We have found that on a small network, like the one that is used here, the first or second hub configuration is very likely to be the best one. The total time limit is set to a large value as the algorithm will finish in a reasonable amount of time.
- m is set to 1 because the network is relatively small. This way, the first solution of the second hub configuration is already close to the optimal solution. This is caused by the local branching constraint.

In Figure 4-4, the performance of the solver and the algorithm are compared for two cases. For case a, which is the "easier" case of the two, the behavior of the two methods is very similar during the time that the algorithm spends on hub configuration 1. This is possibly a consequence of the fact that one of the methods that the GUROBI solver uses to heuristically find solutions is the local branching algorithm, on which our algorithm is based. After approximately 7500 seconds, the stopping criterion is reached and the algorithm starts working on the second hub configuration. For this hub configuration, the best possible solution is about 3% above the best solution for hub configuration 1, which is a consequence of the small network that was used. If a network with more potential hubs is used, the best objective values of the different hub configurations would be closer to each other. We also see that the first solution that the algorithm finds, is close to the best solution that the algorithm finds for this hub configuration. This is because when finding the second hub configuration, a local branching constraint is used that ensures the solution that is found is close to the best solution for hub configuration 1. And since the two hub configurations are similar, the found solution is already of good quality.
For case b, the GUROBI solver outperforms the algorithm during the first 2000 seconds, after that the algorithm is able to find better solutions quicker than the solver, and after that the GUROBI solver finds the best solutions. Furthermore, in both cases we see that the first hub configuration is the best one. This is partly because we let the solver run for a while in stage 1, which ensures the solution that is found is close to the optimum. It is also partly because the number of potential hubs is small here, and so the best solutions for the different hub configurations are far apart, which means it is likely that the solver finds the best one first.

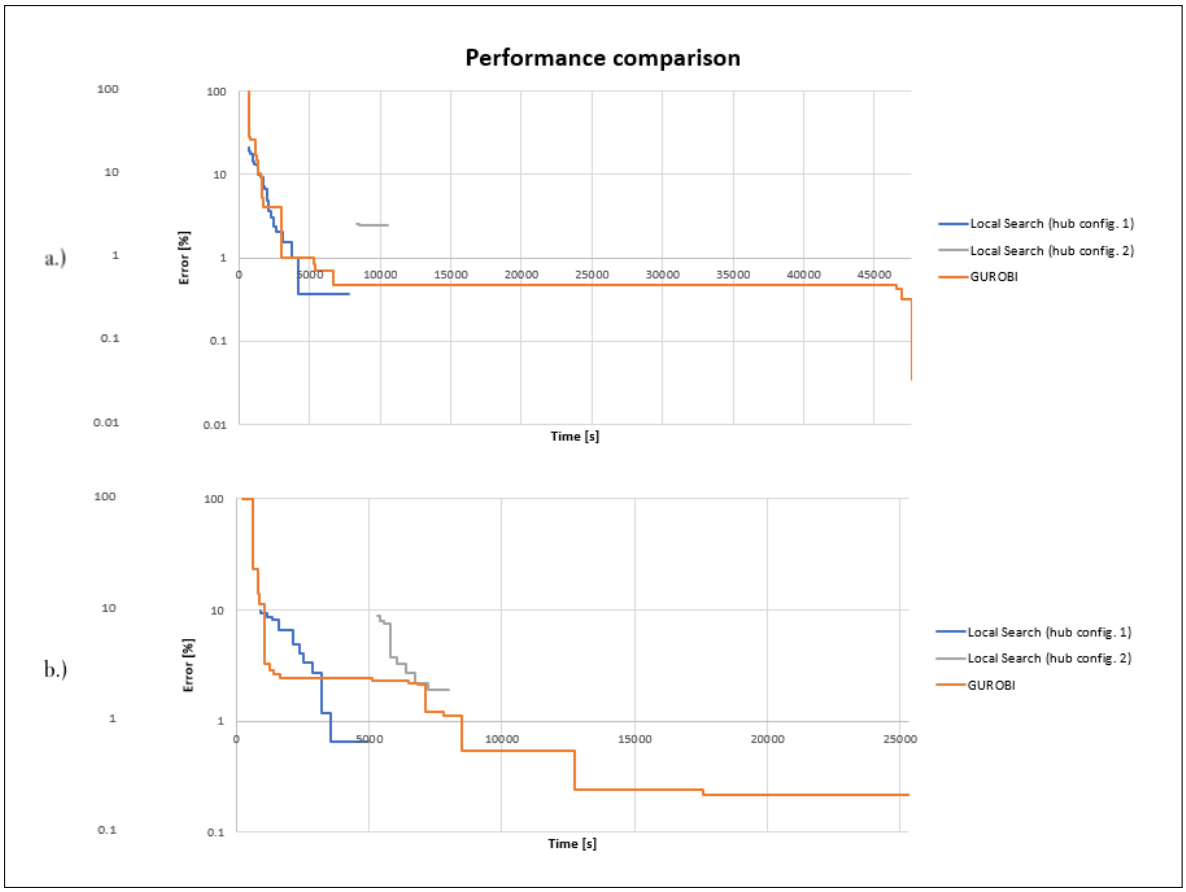


Figure 4-4: Performance of the algorithm for cases *a* and *b*. The error is plotted on a logarithmic scale.

4-6 Conclusion

Because of how the algorithm is designed, the first two stages mainly determine how quickly the algorithm finds its way to the neighborhood in which the optimal solution lies, and the last two stages mainly determine the quality of the found solution.
From the performance comparison we can conclude that the GUROBI solver performs better during the first two stages, and during stages 3 and 4 the algorithm is able to find good quality solutions quicker than the solver. This is why in future implementations, it might be beneficial to use a combination of the two methods; First let the solver find a solution that is reasonably good, and then see if the algorithm can improve upon that solution.

Quality of the found solution Based on the cases that were tested, we expect that the error of the found solution is less than 1%, provided that the algorithm is allowed to complete all stages of the first hub configuration. We expect that this also holds for other cases, as the quality of the final solution is mainly based on the time limit that is set for the solution processes in stage 3 and 4, and these limits are set based on the actual problem. If a problem

is complex, then the time limit is higher than when the problem is easy, and this gives a high chance that a solution with an objective value that is within 1% of the optimum is found. We can also say that the algorithm probably will not find the optimal solution since this would require too much time and this would defeat the purpose of the algorithm.

Settings The following conclusions can be drawn about the high-level settings:

- If a good solution needs to be found within the shortest amount of time possible, the total number of hub configurations should be the dominant setting, and it should be set to 1 or 2. This gives a high probability of a good solution. If a lot of time is available, the time limit should be the dominant setting and it should be set to the actual time that is allowed. Especially in a larger network, it might be beneficial to try extra hub configurations ot see if the best solution can be improved upon.
- We think that in practice, m can be very small or even 1, as the settings of stage 1 ensure that the hub configuration that is found is either the best one, or one that is very close to it.

Chapter 5

Case study

5-1 Introduction to the case study

In this chapter, the findings of the previous chapters are applied to an actual case. The network in this case is larger than the networks that were tested in the previous chapters. Also, the parts of the network that need to be chosen is different. In the case, there are some big customers that send and receive a large quantity of parcels. Furthermore, the hubs in the network are overused, which has a negative influence on multiple aspects such as the punctuality of the parcels. One of the options to resolve this, is to decrease the flow in the network by applying a combination of two measures. The first measure is to no longer transport the parcels of some commodities, and the second is to no longer transport any parcels of outgoing commodities of some of the big customers. For the second measure an extra requirement must hold, which says that if at least one of the outgoing commodities of a big customer is transported, then *all* of them have to be transported. This means there are four decisions that need to be made:

- The parcels of which commodities should be transported; there is a fixed price, and thus market share, for every commodity, and there is a yes/no choice to be made.
- For which of the big customers should the outgoing parcels not be transported.
- Which paths should the commodities use.
- Which arcs should be operated.

Of course these decisions have to be valid with respect to the constraints, such as the hub capacity constraints, and the objective is to maximize the total profit. The implementation of the variables, such that solving the model answers these questions, is shown in Table 5-1. In the comment-field of the market share, the term quasi-binary used, by which we mean that the model allows these variables to assume a value on the continuous range from 0 to 1, but in every optimal solution, the values will be binary. This is explained in the part on linearization in Section 3-4-3. In the problem where, instead of determining the price, there is only a choice to include or exclude a commodity, the MINLP model as it was originally

formulated reduces to a MILP model since the square of 1 is itself. This model formulation could also be used, but relaxing this binary choice yields practically the same solutions, while having a smaller solution time.

Table 5-1: Implementation of variables in the case study.

Variable	Range	Domain	Comment
Market share	Continuous and bounded	Set of potential paths	Also acts as path-choice variable, implemented such that they are quasi-binary
Arcs	Binary	Set of potential arcs	-
Big customers	Binary	Set of potential big customers	-
Hubs	-	Set of fixed hubs	Capacitated

5-2 Solution approach

As mentioned in the introduction, the idea in this chapter is to apply the findings that were presented in the previous chapters. This means that the non-linear model is approximated by a MILP formulation, after which a solution to this formulation is found by using the approach from Chapter 3 and/or Chapter 4.

5-2-1 The MILP formulation

As was the case for Chapter 4, the formulation of the MILP model is very similar to formulation (3-12) to (3-20). The only differences with that formulation are as follows:

- The maximum market share $\dot{x}^{\max}(k)$ of a commodity k is 1 (100%), and the total flows f_k^w and f_k^{pcs} are defined as the flow that is transported in the current situation (where the hubs are overused).
- We define C as the set of big customers, and $P^{O(c)}$ as the subset of P containing all potential paths that have a big customer c as their origin. To ensure that if at least one of the outgoing commodities of a big customer c is offered, all of them are offered, we introduce the constraints:

$$\dot{x}_p \leq z_c \qquad \forall c \in C, p \in P^{O(c)}, \tag{5-1}$$

and

$$\sum_{p \in P^{(c)}} \dot{x}_p \geq z_c \cdot \sum_{k \in P^{(k)}} \dot{x}^{\max}(k) \qquad \forall c \in C, \tag{5-2}$$

where $z_c \in \{0, 1\} \ \forall c \in C$.

Furthermore, the linearization, as described in Section 3-4-2, has one line, and is such that when the market share is 100%, the revenue is equal to the actual modeled revenue in the current situation, i.e. before any of the four decisions are made that were presented in the introduction. As we described in Section 3-4-3, if only one line is used for the approximation, the solution is such that the market share is 0 or 1 for every commodity.

5-2-2 Finding a solution

In Chapter 3 (The Fixed Network), the solution to the MILP model was found by using the GUROBI solver, and, depending on the situation, a combination of the following methods:

- Path relaxation (Section 3-4-2).
- Path pre-selection (Section 3-4-3).
- Arc pre-selection (Section 3-4-3).
- Using GUROBI as a heuristic (Section 3-4-3).

In Chapter 4 (The Variable Network), the solution to the MILP model was found by applying the newly introduced local search algorithm.

In this chapter, we will combine the methods from these chapters. This means we will first use the solver until a stopping criterion is reached, and then we will use the local search algorithm to see if any improvements can be made. Below, we discuss the specifics of how these methods are applied.

Applying methods from Chapter 3 We assume that the error of the objective value of the final solution that is maximally allowed, is 1%. This is based on the assumed precision of the input data, and the goal of the problem, which is strategic in nature. To achieve this error we choose the following settings, which are explained in Section 3-4-3:

- Linearization: 1 line.
- Path relaxation: Yes.
- Path pre-selection: 6 iterations.
- Arc pre-selection: No.
- GUROBI as heuristic: Yes.

The stopping criterion we use, is based on the average time it takes for the solver to find a better solution. The solver continually finds better solutions and the time between these improvements is averaged. If, at a certain moment in time, the time that has passed since the last improvement, reaches more than 3 times this average value, the solver is stopped. This ensures that the "quick" improvements are done, but when the solver cannot easily find a better solution, the local search algorithm takes over from the best solution that was found. In the Figure 4-4*a*, this would be after approximately 9000 seconds.

Applying methods from Chapter 4 In the computational experiments, which are discussed in Section 4-5, we have seen that when a suitable stopping criterion is used, the local search

algorithm can potentially improve the solution, and this improvement would require less time than letting the solver continue its solution process. Since at this point, the quality of the solution is already relatively good, only stages 3 and 4, as they are shown in Figure 4-1, are performed. Since the only settings that need to be manually set for the algorithm concern the search for hub configurations, and in this situation, no new hub configurations need to be found, we do not need to specify any settings here.

5-3 Solution process

The solution process is depicted in Figure 5-1. After 147,497 seconds, the solver finds the optimal solution, and after 293,485 it proves that it is the optimal solution. The figure confirms the conclusion of Section 4-6, which says that the local search algorithm can improve the best found solution of the solver at the moment it has found a reasonably good solution. When we compare the time it takes until the algorithm terminates (and returns a solution), and the time it takes the solver to find a solution that is equally good or better, we see that the algorithm is 38,914 seconds, or 65.3%, quicker. After this however, no better solution can be found by the algorithm in its current configuration, while the solver will continue to find better solutions.

We expect that the performance of the combination of the solver and the algorithm can be further improved, for instance by changing the internal parameters such that the algorithm runs later in the original solution process. But since the aim was to design an algorithm that can be used on different cases without much parameter tuning, this has also not been done on this case.

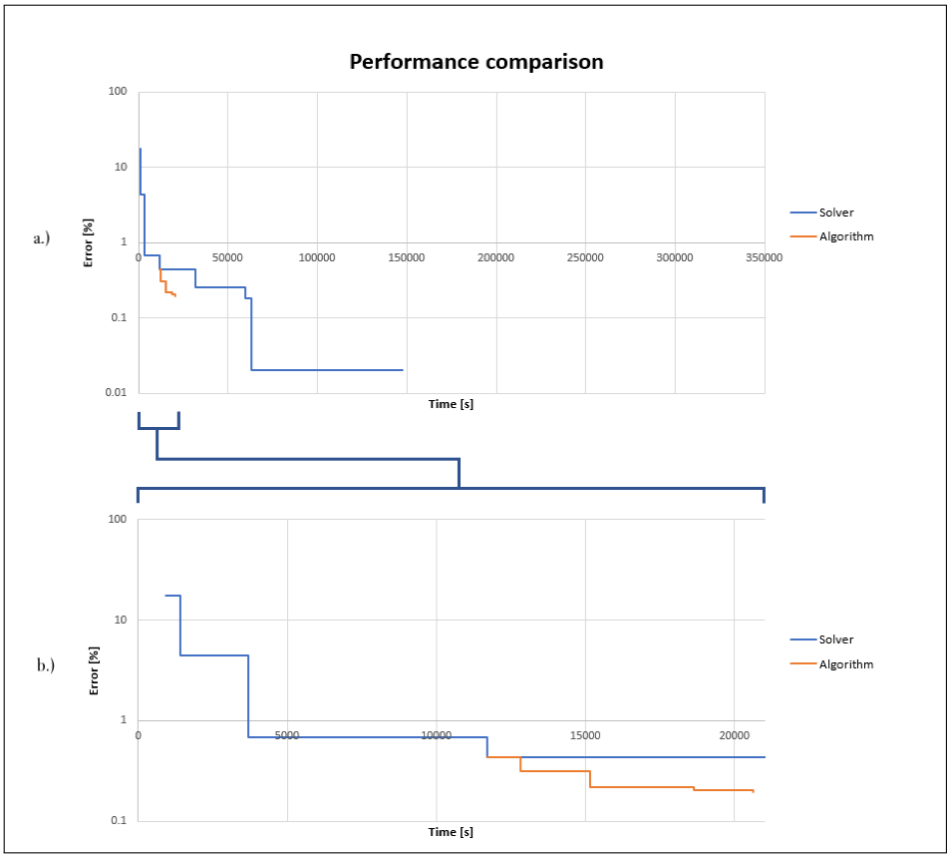


Figure 5-1: Comparison of the solution processes of the algorithm and the solver. The only difference between *a* and *b* is the time domain. The error is plotted on a logarithmic scale.

Chapter 6

Conclusions and further research

In this chapter, the findings of this thesis are presented. First, the research question is answered, followed by some conclusions that are also applicable to a wide range of similar problems, then some limitations of the research are discussed, and finally some suggestions for further research are proposed.

6-1 Research question

We begin by restating the research question:

Which solution approach is most suitable for finding the optimal pricing strategy and network structure in Situations 1 and 2, when maximizing the total profit is the objective? And which conclusions can we draw regarding the computational complexity?

1: "The Fixed Network"	The number of hubs and their locations are fixed and they have a fixed capacity, depots have a fixed location and cannot be closed, and arcs are allowed to be opened or closed.
2: "The Variable Network"	The number of hubs, their locations, and their capacity need to be chosen. Depots have a fixed location, and both the depots and arcs are allowed to be opened or closed.

In both situations, we conclude that the best way to approach the problem, is to formulate one mathematical programming model that includes all aspects of the problem, and then to approximate this by a MILP model. This approximation can best be done using a linearization approach that approximates the non-linear objective function by linear functions. This model is then solved, where depending on the error and the solution time that are required, and whether a guarantee on the error is needed, a combination of several methods and parameters can be chosen. The methods and parameters consist of the parts described in Section 3-4-3, and the local search algorithm that is described in 4-4. All methods (including the algorithm) can be applied to both situations, and the way they are implemented requires a

trade-off between the error and the solution time.

With regard to the computational complexity, we start by mentioning the performance variability within instances of MILP models. This is shortly discussed in Section 2-1, and the phenomenon decreases the reliability of the results, although we expect that this effect is small. Another important factor is performance variability between instances, which is notoriously high for **NP-Hard** MILP models. This is why we have tried to substantiate all statements in this conclusion, on the performance of the solution of models, as much as possible by extensive testing. But performance on individual instances might still deviate from the described results.
Regarding the computational complexity of the situations of the research question, we expect that when the network is small (around 20 depots, 4 hubs, 1000 commodities, and fully connected) and we look at Situation 1, the resulting MILP model is solvable in a reasonable amount of time, a few hours on a standard machine, and that the error that is caused by the applied approach is less than 0.5%. When we look at Situation 2, the model for a small network is not solvable to optimality, and this results in a total error of approximately 1%. The methods were not extensively tested on larger networks, but we imitated this by including a large number of commodities and possible paths. We also applied the methods to a real case that is based on a larger network (Chapter 5). Based on the results, we expect that for Situation 1, an error between 1% and 2% is possible for networks that have at most 5 times more depots and commodities than the tested small network, and in which the hub capacities are not too low. For larger networks and Situation 2, the error is higher, harder to predict, and a guarantee on the quality is not possible, but nonetheless we expect it to be less than 3%. We do not expect that these errors will be a problem, as the accuracy of the input data for strategic decisions, particularly the price-market share relation, is usually much lower.

6-1-1 Similar problems

There is a large range of problems that are similar to the problems stated in the research question. Examples of these MILP network optimization problems are the real case from Chapter 5, and problems where only the sizes and locations of the hub need to be chosen. In finding a solution approach for these kind of problems, the exact formulation is very important. To estimate which aspects of the problem can be included in the model before solving it will take too long, the impact of including different parts is shown in Table 6-1. Since the interdependence between the parts and the actual input data also greatly affects the solution time, we only give an indication for each part. This is based on the multitude of cases that were analyzed during the work on the thesis. For the impact of the different parts, we assume that the decision to open or close an arc, and the decision on which paths are chosen, will always be included in the model. The impact of these parts is covered in Section 3-4-4. The way the parts in the table are implemented is described in Sections 3-2, 4-2, and 5-2-1.

Table 6-1: An indication of the impact of including several aspects in a MILP network optimization model. If the number is high, the impact is high, and vice versa.

Part		Impact factor
Price decision[(1)]	Continuous range	50
	Quasi-binary	10
Depot decision[(2)]		3
Hub decision[(3)]	High	2
	Medium	6
	Low	25
Area decision[(4)]		50
Hub capacity constraints[(5)]	High	2
	Medium	8
	Low	30
Balancing constraints[(6)]		1.5
Outgoing-commodities constraints[(7)]		1.5

(1) Including the price decision, either from a continuous range, or quasi-binary (see Section 5-1).
(2) Including the decision to open or close a depot.
(3) Including the decision to open or close a hub, and which of three possible capacities it should have. The average values of these capacities can be high, medium, or low.
(4) Including the decision to assign every area to a depot. For this, different input data is needed, such as the flows on the area-level and PUD-costs.
(5) Including the hub capacity constraints, which can be high, medium, or low.
(6) Including the constraints that ensure that either all outgoing commodities of a depot are open, or all of them are closed.
(7) Including the constraints that ensure that if at least one outgoing commodity of a depot is transported, all of them are transported (see Section 5-1).

What follows next are some conclusions that are applicable to both the research question and the range of problems that is described in Table 6-1:

- When we look at the impact of the input data (the price, total flow, and hub capacities) on the solution time, the biggest factor is the capacities of the hubs. When they are such that in the optimal solution, several of them are at their full capacity, the model is very hard to solve. However, we also know that the model will never be infeasible, which is what happens when the hub capacities are too low in problems where the prices (and thus also the flows) are fixed.

- The difference in solution time between using 6 or 30 lines for the linearization, is a factor 10. And this relation is approximately linear. The error in this domain decreases logarithmically from 1% to 0.25%. We expect that these general trends are the same for similar problems.
- Since for none of the cases that were solved during this thesis, relaxing the constraints that ensure one and only one path is chosen per commodity, into constraints that ensure at most one path is chosen per commodity, made a significant difference, we expect this to be the case for all similar problems.
- Defining constraints for MILP models in different ways can have counter-intuitive effects on the solution time. This we have seen for the sum-constraints and the one-by-one-constraints in Section 3-4-6, where we proved that the sum-constraints are stronger, but solving the model needs less time when using the one-by-one-constraints. As a general rule, it is better to have a large amount of constraints, each containing few variables, than vice versa. But for future projects, we recommend to always test this in practice.
- The same holds for the input data. This we have seen in for example Section 3-4-4 for the number of potential paths that are included in the model.

Local search The best way to use the local search algorithm, is to combine it with the "regular" solution process of a MILP solver; First let the solver find a solution that is reasonably good, and then see if the algorithm can improve upon that solution. When this combination is used on a problem, the quality of the found solution is unknown to a large degree, and this makes it unfit to use in some situations. However, in the following situations, the exact quality of the solution is less important and the algorithm can be of great use:

- In an exploratory investigation into the possible benefits of including the revenues in a network optimization problem.
- A model with a large quantity of possible hub sizes is very hard to solve. An alternative is running multiple models with fewer sizes to determine the most promising sizes, which can then be used in a final optimization. For running the models with fewer sizes the local search procedure can be used.
- For large and difficult models, a model might be used in an iterative process in conjunction with other optimization models or a procedure that updates the input data. Here the local search algorithm might also be of use.

Furthermore, the principle of constraining the problem such that solutions are close to a current solution, can be used in a multitude of realistic situations in which large alterations of the current network are not allowed.

6-2 General conclusions

There are three important general conclusions that we have drawn from the work on the thesis.

- The work on the problem definition and the subsequent model construction and solution finding, has shown that these phases cannot be done separately. The actual requirements

and objectives of the problem, and the required solution time and error have a lot of mutual dependencies, and should therefore be defined iteratively.
- From analyzing many solution processes of MILP models, we conclude that the process of finding increasingly good solutions is similar for many of the more difficult problems (see for example Figure 4-4-a). After the first phase of the process, the solution is within a relatively small margin from the optimal solution, and after this, finding the optimal solution takes a large amount of time. This knowledge can be exploited for many similar problems.
- Because MILP solvers are black-box algorithms, discovering their workings can only be done by using different input and analyzing the output. Although this gives great insight for many general aspects, investigating the more detailed aspects often leads to unpredictable results.

6-3 Limitations

Here, a few limitations of the thesis are described that were not already mentioned in the previous parts of this chapter.
The first limitation is the requirement that the price market-share relation should be concave and non-increasing if the linearization approach is used. We expect that this is only a minor limitation as most actual relations will have these characteristics. If a relation should not be concave, extra binary variables will need to be introduced in the model, or the discretization approach needs to be used, and this increases the solution time significantly. Furthermore, the non-linear terms in the objective function need to be univariate. If this is not the case, for instance when the market-share of a commodity also depends on the market-shares of other commodities, the linearization approach cannot be used.
Another limitation is that for all cases in the thesis, the prices can be chosen for every commodity, and that they are always a function of the distance between the depots, but this might not be how prices are set in reality. It could be that they are the same for all OD-combinations, as is the case for most express service providers in the Netherlands. Or it could be that the prices are set per origin and destination region, and that they do not depend on the distance between the regions but on the prices of the competition. When the prices are based on the distance and chosen for every commodity, as is the case in the thesis, many commodities will have revenues and costs that are close to each other, making it hard to determine whether they are profitable. If this is not the case, as in the two examples that were just mentioned, the revenue and the cost will be further apart. This is why we expect the complexity of the model to be lower in these cases.

6-4 Further research

In this section, some suggestions are done for further research. They are mainly based on ideas that were conceived during the work on the thesis. First some suggestions are presented

that are improvements of the MILP models that could be used in the thesis:

- The pre-selection of arcs and paths is based on the solution of the LP-relaxation, and the stronger the formulation is, the closer the optimal solution is to the optimal solution of MILP model, and the better the selection of arcs and paths is. Currently, the original MILP formulation is used, but a general purpose MILP solver generates cuts that strengthen the formulation, and this could be used to increase the quality of the pre-selected set of arcs and paths.
- The choosing of lines for the linearization is currently done in a way that ensures the average distance between the non-linear function and the linear function is minimized over the entire domain. However, we can fairly accurately predict in which part of the domain the optimal market-share will lie, namely the part where the revenue is maximal in the isolated case of optimizing only over the revenue and the variable cost. Choosing the linearization such that the distance between the functions is minimal here, will likely decrease the error.
- In modern general purpose MILP solvers, it is possible to feed a solution s to the solver that it then uses as a lower bound in the B&C tree. This speeds up the solution process, and it could be beneficial to implement in both the general solution process as used in Chapter 3, and the local search procedure as used in Chapter 4. In the general solution process, the local search procedure could be used to generate s, and in the local search procedure s could be the solution found in the previous iteration.

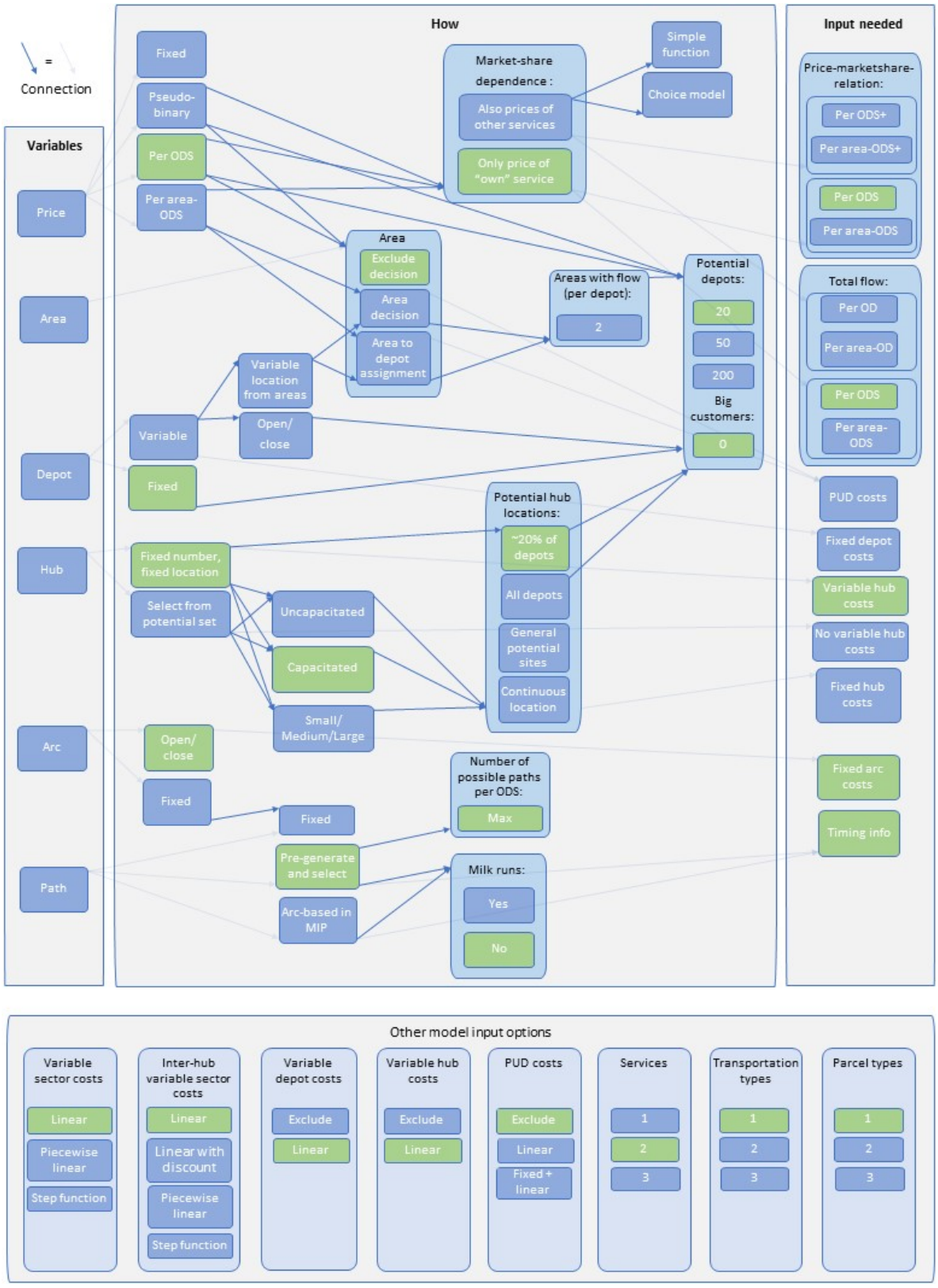
Next, we discuss two suggestions that are not only extensions of the research question, but show possibly interesting directions for further research.

- In every optimal solution to network optimization model, there are inter-dependencies between variables that are based on problem specific information. An example could be that when a certain hub is open, some arcs are always open because they are the cheapest way to connect the hub to a part of the network. Another example is that when a certain arc is open, some paths are always chosen because the arc has low variable costs. Combining these inter-dependencies into composite variables could decrease the solution time. However, this method is effectively a way to cut off a part of the feasible region, and this should be done with great care. Also, a guarantee that good solutions are not cut off cannot be given, making this a heuristic approach.
- Robust optimization (Ben-Tal et al. (2009)), is constraint-wise approach to incorporate uncertainty into an optimization model in a fairly easy way, which potentially does not require a large amount of extra solution time. Modern robust optimization deals primarily with non-probabilistic models of uncertainty, meaning it defines a possible perturbation of parts of the input data, and this allows a worst/best case analysis. Disadvantages are that it can lead to overly conservative solutions, and that potential information on the distribution of the uncertainty is not used. However, the uncertainty in longer term predictions makes it insightful to include robust optimization in strategic decisions like the problems in this thesis.

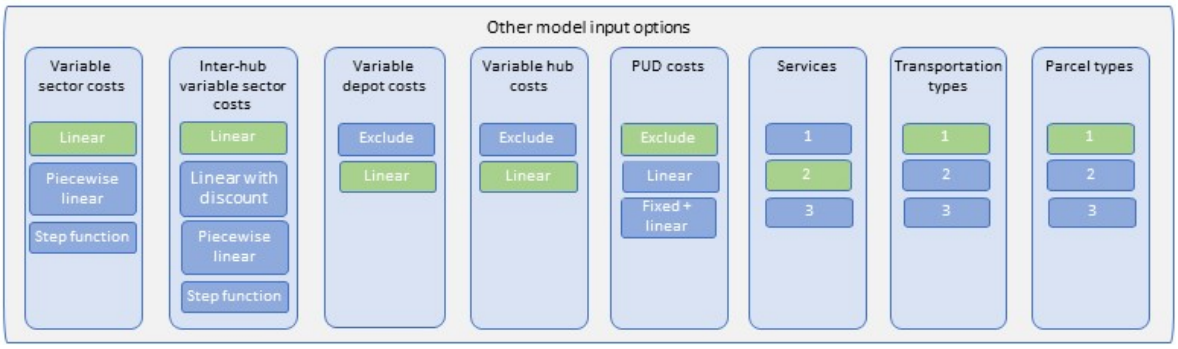
Appendix A

Problem overview

A-1 The Open Network



A-2 The Variable Network



Bibliography

Armacost, Andrew P., Cynthia Barnhart, Keith A. Ware, and Alysia M. Wilson (2004), "Ups optimizes its air network." *European Journal of Operational Research.*

Balas, Egon (1979), *Annals of Discrete Mathematics, Discrete Optimization II*, chapter Disjunctive Programming, 3 – 51. Elsevier.

Ben-Tal, Aharon, Laurent El Ghaoui, and Arkadi Nemirovski (2009), *Robust Optimization.* Princeton University Press.

Benders, J.F. (1962), "Partitioning procedures for solving mixed-variables programming problems." *Numerische Mathematik.*

Brotcorne, L., F. Cirinei, and G. Marcotte, P. Savard (2011), "An exact algorithm for the network pricing problem." *Discrete Optimization.*

Camargo, Ricardo Saraiva de, Gilberto de Miranda, and Arne Lokketangen (1998), "Hub location with flow economies of scale." *Transportation research.*

Campbell, James F. (1994), "Integer programming formulations of discrete hub location problems." *European Journal of Operational Research.*

Campbell, James F. and Morton E. O'Kelly (2012), "Twenty-five years of hub location research." *Transportation Science.*

Castro, Jordi (2003), "Solving difficult multicommodity problems with a specialized interior-point algorithm." *Annals of Operations Research.*

Contreras, Ivan, Jean-Francois Cordeau, and Gilbert Laporte (2011), "Benders decomposition for large-scale uncapacitated hub location." *Operations Research.*

Danna, E. (2008), "Performance variability in mixed integer programming. presentation, workshop on mixed interger programming." URL `http://coral.ie.lehigh.edu/Ëœjeff/mip-2008/talks/danna.pdf`. Accessed: 12.07.2018.

Dobson, Gregory. and Phillip J. Lederer (1993), "Airline scheduling and routing in a hub-and-spoke system." *Transportation Science*.

Eiselt, H.A. and Vladimir Marianov (2008), "A conditional p-hub location problem with attraction functions." *Computers and Operations Research*.

Firat, M., N.P. Dellaert, and W.P.M. Nuijten (2012), "Solving routing problems by exploiting the dual of a master lp formulation." Technical report, Technische Universiteit Eindhoven.

Fischetti, M. and A. Lodi (2003), "Local branching." *Mathematical Programming*.

Gelareha, Shahin, Stefan Nickel, and David Pisinger (2010), "Liner shipping hub network design in a competitive environment." *Transportation research*.

Gendreau, Michel and Jean-Yves Potvin (2010), *Handbook of Metaheuristics*. Springer Publishing Company, Incorporated.

Gendron, Bernard (2011), "Decomposition methods for network design." *Procedia - Social and Behavioral Sciences*.

GUROBI Optimization (2018), "Algorithms in gurobi." URL `http://www.gurobi.com/pdfs/user-events/2016-frankfurt/Die-Algorithmen.pdf`. Accessed: 21.02.2018.

Gurobi Optimization (2018), "Gurobi optimizer quick start guide for windows." URL `http://www.gurobi.com/documentation/7.0/quickstart_windows/index.html`. Accessed: 09.07.2018.

GUROBI Optimization (2018), "Mixed integer programming." URL `http://co-at-work.zib.de/files/Gurobi_MIP.pdf`. Accessed: 25.0.2018.

Hijazi, Hassan, Pierre Bonami, Gerard Cornuejols, and Adam Ouorou (2012), "Mixed-integer nonlinear programs featuring "on/off" constraints." *Computational Optimization and Applications*.

IBM Corporation (2018), "Ibm ilog cplex optimization studio cplex user's manual." URL `https://www.ibm.com/support/knowledgecenter/SSSA5P_12.6.2/ilog.odms.studio.help/pdf/usrcplex.pdf`. Accessed: 09.07.2018.

Iri, Masao (1996), *System Modelling and Optimization*, chapter Network flow - Theory and applications with practical impact. The International Federation for Information Processing.

Kallrath, J. (2004), *Modeling Languages in Mathematical Optimization*. Springer US.

Kim, Daeki, Cynthia Barnhart, Keith Ware, and Gregory Reinhardt (1999), "Multimodal express package delivery: A service network design application." *Transportation Science*.

Kuiteing, Aime Kamgaing, Patrice Marcotte, and Gilles Savarda (2015), "Network pricing of congestion-free networks: The elastic and linear demand case." *Management Science*.

Labbe, Martine, Patrice Marcotte, and Gilles Savard (1998), "A bilevel model of taxation and its application to optimal highway pricing." *Management Science*.

Lin, Ming-Hua, John Gunnar Carlsson, Jianming Ge, Dongdong Shi, and Jung-Fa Tsai (2013a), "A review of piecewise linearization methods." *Mathematical Problems in Engineering*.

Lin, Ming-Hua, John Gunnar Carlsson, Dongdong Ge, Jianming Shi, and Jung-Fa Tsai (2013b), "A review of piecewise linearization methods." *Mathematical Problems in Engineering*.

Luer-Villagra, Armin and Vladimir Marianov (2013), "A competitive hub location and pricing problem." *European Journal of Operational Research*.

Mahmutogullari, Ali Irfan and Bahar Y. Kara (2015), "Hub location under competition." *European Journal of Operational Research*.

Marchand, Hugues, Alexander Martin, Robert Weismantel, and Laurence Wolsey (2001), "Cutting planes in integer and mixed integer programming." *Discrete Applied Mathematics*.

Meuffels, W. J. M. (2010), *The Design of Road and Air Networks for Express Service Providers*. Ph.D. thesis, Tilburg University.

Mittelmann, H. (2018), "Mixed integer linear programming benchmark (miplib2010)." URL `http://plato.asu.edu/ftp/milpc.html`. Accessed: 23.01.2018.

Nemhauser, George L. and Laurence A. Wolsey (1988), *Integer and Combinatorial Optimization*. Wiley-Interscience.

O'Kelly, M. E. and D. L. Bryan (1998), "Hub location with flow economies of scale." *Transportation research*.

O'Kelly, Morton E., Henrique Pacca L. Luna, Ricardo S. de Camargo, and Gilberto de Miranda Jr. (2015), "Hub location problems with price sensitive demands." *Networks and Spatial Economics*.

Salehi, F., L. Ryssel, and J. Matuska (2015), "Europe's CEP market: Steady growth begins to shift." A.T. Kearney Inc.

Glossary

List of Acronyms

PUD	Pick Up and Delivery
OD	Origin-Destination
CEP	Courier, Express and Parcel
HS	Hub and Spoke
NLIP	Non Linear Integer Programming
MILP	Mixed Integer Linear Programming
MINLP	Mixed Integer Non Linear Programming
NLP	Non Linear Programming
QCP	Quadratically Constrained Programming
QP	Quadratic Programming
IP	Integer Programming
LP	Linear Programming
NPP	Network Pricing Problem
MIQP	Mixed Integer Quadratic Program
SOCP	Second-Order Cone Programming
B&C	Branch And Cut
B&B	Branch And Bound
OIO	Ortec Infrastructure Optimizer
AIMMS	Advanced Interactive Multidimensional Modeling System
B2B	Business-To-Business
B2C	Business-To-Consumer
MIR	Mixed Integer Rounding
VNS	Variable Neighborhood Search
MIR	Mixed Integer Rounding
LNS	Large Neighborhood Search
OR	Operations Research