Technische Universität München

Fakultät für Mathematik
Lehrstuhl für Angewandte Geometrie und Diskrete Mathematik

Heuristic for multi-echelon facility location problems with non-linear inventory considerations

An evolutionary approach to discrete facility location problems

A master's thesis by Christoph Bolkart

Examiner:	Prof. Dr. Peter Gritzmann
Advisor:	Wolfgang F. Riedl, Dr. Michael Ritter
Submission Date:	September 29, 2014

I hereby confirm that this is my own work, and that I used only the cited sources and materials.

München, September 29, 2014

Christoph Bolkart

Abstract

This thesis aims to extend ”facility location - allocation” problems in three directions. Multiple products are considered, a second echelon of distribution centers is added, and the non-linear inventory costs are included in objective function. Solving such problems with linear optimization techniques proves to be too computationally expensive. Therefore, a heuristic approach, based on genetic algorithms, will be developed. In the heuristic approach a large number of non-linear flow problems are solved, for which a simple and efficient algorithm will be formulated. Using this heuristic, large real world problems can be solved in relatively short time.

Zusammenfassung

In dieser Thesis wird das bekannte ”Facility Location - Allocation” Problem in drei Richtungen erweitert. Mehrere Produkte werden betrachtet, eine zweite Lagerstufe wird hinzugefügt und nicht-lineare Bestandskosten werden zur Zielfunktion hinzugefügt. Die Lösung eines solchen Problems mit Hilfe von Methoden der linearen Optimierung benötigt zu viel Rechenzeit. Deswegen wird eine Heuristik in Form eines genetischen Algorithmus entwickelt. Die Heuristik muss hierbei eine große Anzahl nicht-linearer Flussprobleme lösen, für die ein einfacher und effizienter Algorithmus formuliert wird. Mit dieser Heuristik konnten große reale Probleme in relativ kurzer Zeit gelöst werden.

Contents

Chapter 1

Introduction

Deciding on the location of facilities to provide optimal service has been a problem for a long time. [LMW88] write that in most American cities, fire stations were located approximately 20 blocks apart, because it had been observed, that ”horses pulling a fire wagon ... can run no farther than ten city blocks”.

In the current business environment customers demand ever shorter lead times and a very high service level. In business-to-business relationships the higher requirements are an outcome of the introduction of just-in-time production. In business-to-customer interactions the very good service level of companies such as Amazon has created an expectation among customers, that products should arrive the next day. To keep customers and create growth, businesses must adapt to these requirements and find ways to meet them. Thus, the main task of supply chain management is to enable a company to reliably and quickly deliver the products to the customers who want them at a low cost.

The term ”Supply Chain Management” was coined by Keith Oliver, a consultant from Booz Hamiltion, in an 1982 interview with the Financial Times ([Krane]). According to [TG96] the term Supply Chain Management describes ”the management of material and information flows both in and between facilities such as vendors, manufacturing and assembly plants and distribution centers”.

In the literature, e.g. [Ste89], a distinction is frequently made between three levels of planning and operation, which are the strategic, the tactical and the operational level.

- On the strategic level long term decisions are taken with a time horizon of several years. According to [SLKSL04] ”the strategic level deals with decisions that have a long-lasting effect on the firm. These include decisions regarding the number, location and capacities of warehouses and manufacturing plants, or the flow of material through the logistics network”.

- The tactical level deals with questions about how to conduct business within the strategic framework. ”It involves translating the strategic objectives and policies into complementary goals and objectives for each function to provide balance to the supply chain. The functional goals provide the drivers for achieving the balance and inventory, capacity and service are the levers by which balance is achieved." [Ste89]

- The operational level, according to [Ste89], "is concerned with the efficient operation of the supply chain" and mainly deals with short term decisions.

Strategic decisions in the realm of supply chain management often touch the design of the supply chain. During the process of Supply Chain Design decision makers "determine the location, size and optimal numbers of suppliers, plants and distributors to be used in the network" according to [SR02]. They say that on the strategic level Supply Chain Design deals with "determining the nodes and arcs of the SCN[1] and their relationships". Supply chain design "is critical, strategic and inherently complex" according to [MND14]

The goal of supply chain design processes is to find an optimal solution considering the necessary trade-offs between

- Low cost
- High customer service
- High responsiveness to changes in the environment
- High robustness in case of interruptions

In unstable regions with dynamic growth businesses value a supply chain which can quickly adjust to interruptions or fast-growing demands. However, since most supply chain design projects focus on stable regions like Western Europe and North America, the key differentiating factors are cost and customer service.

Due to the increasing service requirements and cost pressure, many companies face the challenge of designing an adequate network to distribute their products from their plants to their customers. Procuring all raw materials required for the production of a product and going through all production steps takes time. Consequently, for most products it is impossible to start production only after a customer order has been received. This means, that production plans must rely on forecasts. The quantities cannot be shipped directly to the customers, because they may not have ordered the product yet.

Furthermore, it is unknown, if and when a customer will order a certain product, because customer demand is stochastic and fluctuates over time. Thus, the produced quantity must be accessible for a wide set of customers, who could ask for the product.

The inflexible production and the unpredictable demands make it necessary that companies operate distribution centers at which they store their finished products. Most

[1]Supply Chain Network

of the times a customer orders a product, he receives it from one of these distribution centers.

Since the fix cost of a distribution center is rather high, companies avoid operating too many. However, the fewer distribution centers a company operates, the larger the distances to their customers becomes. The customers then have to wait for a long time until they receive their products. Furthermore, small individual shipments are more expensive than larger ones. Sending 30 individual parcels from Germany to customers in the Madrid area is more expensive than sending one truck to a hub location in Madrid and having the parcels delivered from there.

As described in the last paragraphs, several factors, such as the operating costs and inventory reduction, push for fewer distribution centers and more centralization. On the other hand, transportation costs and the service level are drivers for a less centralized structure and more distribution centers. A middle ground must be found when deciding where to locate distribution centers.

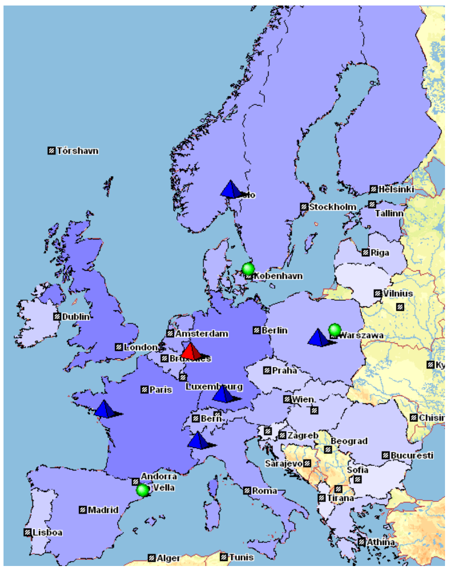
In figure 1.1 two different fictitious network configurations are depicted. In 1.1a a very centralized scenario with one central distribution center and three regional ones is presented. This stands in stark contrast with 1.1b where ten regional distribution centers are distributed across Europe. Blue pyramids indicate factory locations, the red pyramid indicates the location of the central distribution center (CDC) and the green balls represent the regional distribution centers (RDC). Countries with higher sales volume are shaded in a darker blue.

When deciding on the network structure several key questions must be answered.

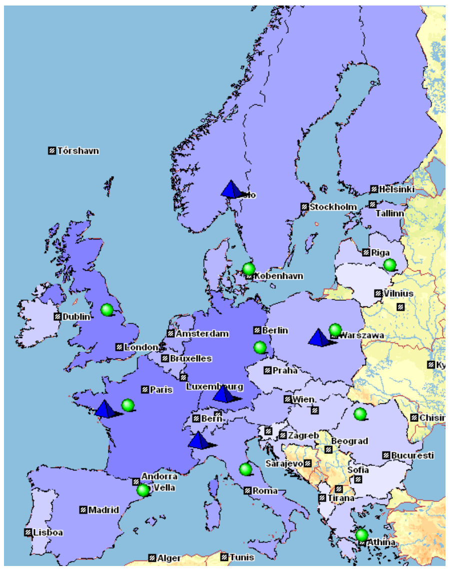
- Should a central distribution center supply the regional distribution centers or should the RDCs receive everything directly from the factories?
- How many regional distribution centers should be opened and where?
- Which products should be stored at the regional distribution centers and which should be stored centrally?
- Which distribution center should supply a customer and can it supply him in an acceptable time?

Being able to find good answers to these questions allows companies to provide good service to their customers at a low cost and compete in demanding environments. The goal of this thesis is to develop a method to find these answers.

The remainder of this thesis is structured in the following way. In chapter 2 an overview of the development of facility location problems and of recent research results will be provided. Chapter 3 will describe the theory behind inventory analysis. The prerequisites and assumptions for the model will be detailed in chapter 4 and the mathematical model will be formulated in chapter 5. Chapter 6 will outline the general



(a) Central network with one central distribution center (CDC) and three regional distribution centers (RDC)



(b) Regional network with no CDC and ten RDCs

Figure 1.1: Blue pyramids represent factories, red pyramid indicates the central distribution center and green balls stand for regional distribution center

solution method, comprised of an evolutionary part and an approach to solve the resulting non-linear flow problems. The evolutionary aspect of the algorithm will be described in chapter 7 and the solution of the flow problems will be discussed in chapter 8. In chapter 9 the computational results will be presented and discussed. Finally, a conclusion of the work will be derived and future research questions formulated in chapter 10.

Chapter 2

Literature review

A vast amount of literature has been written about facility location problems. Recently extensions of the traditional facility location - allocation problem have been formulated tackling the integration of tactical aspects like production, sourcing and inventory and operational aspects like routing alongside the strategic decision about the location of warehouses and distribution centers. Surveys about the incorporation of such aspects into the literature have been written by [BAF12], [KD05], [MNG09], [ŞS07] and [She07].

In the next section an overview of the basic facility location models will be provided and some of the early solution approaches will be presented. In section 2.2 multi-echelon facility location problems will be discussed and a classification for such problems will be reviewed. Finally, in section 2.3 the relatively recent incorporation of inventory costs into the facility location problem will be studied.

2.1 First approaches to facility location problems

The facility location problem (FLP) was introduced by [WP09] in the early 20th century. They describe the problem of placing one warehouse in the 2-dimensional plane with customers located across the plane. The objective was to find a warehouse location $x \in \mathbb{R}^2$ so as to minimize the total distance from the warehouse to n customers located at points $x_i \in \mathbb{R}^2$ with $i \in \{1, 2, ..., n\}$. Weber assumed that the warehouse could be located anywhere in the plane. The resulting problem can be formulated as

$$\min_x \sum_{i=1}^{N} d(x, x_i), \text{ s.t. } x \in \mathbb{R}^2 \tag{2.1}$$

This problem is commonly referred to as the Steiner-Weber problem. However, the first time such a problem was formulated was as early as 1638, when René Descartes posed it to Pierre de Fermat according to [Enc]. A geometric solution for the special case of a triangle was found a few years later by Torricelli. For cases with more than $n \geq 5$ points there exists no exact algorithm [Baj88].

The first iterative algorithm to solve the Steiner-Weber problem was published in 1937 by [End37] in an article called ”Sur le point pour lequel la somme des distances n donnes est minimum” in the Japanese ”Tohoku Mathematical Journal”. However, as [Vaz02][2] mentions in his autobiographical book ”Which door has the Cadillac” he was only interested in proving a theorem.

> Here was a theorem to prove: Does the process converge? Yes it does! And that was what the Tohoku article was about.

He even admits that at the time ”the word algorithm was unknown to [him] and most mathematicians”. Since the algorithm was published in French in a Japanese journal and appeared to be of little relevance, it was quickly forgotten. In the late 50s and early 60s it was independently rediscovered by [Mie58] and [KK62].

Limitations in Weber’s model led to extensions of the problem. In realistic scenarios not all locations on the map are suitable for a warehouse. Reasons could be an extremely high price of land, e.g. a location in the center of an expensive metropolitan area, or a lack of transportation options, because the chosen location lies in a remote area. Therefore it is advisable to use the detailed expert knowledge of supply chain designers about existing transportation infrastructure. Almost all logistics service providers have their main European hubs in the same regions. From these regions it is possible to send products in a very short time to many locations. For example, Amazon has selected Leipzig as a distribution center, because their logistics service provider, DHL, has a large hub there. The same is true for their distribution centers located in Graben, Bavaria, and Rheinberg, North Rhine-Westphalia ([Ama]).

Moreover, companies do not only operate one warehouse and supply all their customers from there. Rather, they use multiple facilities to ensure the satisfaction of lead time requirements and to manage the risk of disruptions. While Steiner-Weber problems have been extended to allow the selection of multiple locations, the focus of research on facility location problems has shifted towards discrete problems.

In discrete facility location problems two strongly related questions are answered simultaneously.

a) Which warehouses, out of a subset of potential warehouses, should be opened?

b) From which warehouse should customers receive their products?

The objective of the discrete facility location problem is to minimize the sum of the warehouse costs and the transportation costs. An example of such a set of locations can be seen in figure 2.1. Five major German cities are represented as nodes in the graph. In discrete facility location problems a subset of the nodes is selected and at these locations warehouses are opened.

[2]Endre Weiszfeld later in life changed his name to Andrew Vaszonyi

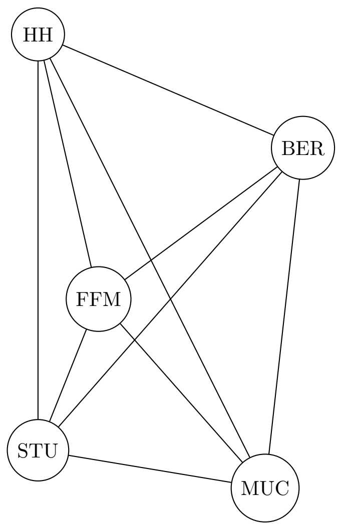


Figure 2.1: Network containing 5 nodes which represent customer cities. Warehouse should be located at one of the nodes

In their seminal works [Coo63] and Hakimi [Hak64], [Hak65] introduced the concept of graphs to facility location problems. Hakimi studied the problem of selecting nodes in a graph to act as switching centers in communication networks or as police stations in a location model. They formulated the following problem.

Define a set of customers C with demand $d_i,\ \forall i \in C$, a set of warehouses W with opening cost $w_j,\ j \in W$ and a unit transportation cost c_{ji} between a warehouse location $j \in W$ and a customer $i \in C$. Let the binary decision variable $x_j = 1$ if location j is opened and let y_{ji} the flow along the edge from j to i. Assume M sufficiently large, i.e. $M \geq \sum\limits_{i \in C} d_i$.

$$\min\ z = \sum_{j \in W} (x_j w_j + \sum_{i \in C} y_{ji} c_{ji}) \tag{2.2a}$$

subject to

$$\sum_{j \in W} y_{ji} \geq d_i \qquad \forall i \in C \tag{2.2b}$$

$$\sum_{i \in C} y_{ji} \leq M x_j \qquad \forall j \in W \tag{2.2c}$$

$$x_j \in \{0, 1\} \qquad \forall j \in W \tag{2.2d}$$

$$y_{ji} \geq 0 \qquad \forall j \in W,\ \forall i \in C \tag{2.2e}$$

This problem is called the Uncapacitated Facility Location Problem (UFLP), because the warehouses do not have a maximum capacity. Replacing M with a maximum capacity value $M_j < \sum_{i \in C} d_i$ means that some warehouses do not have sufficient capacity to satisfy all demands. The resulting problem was called the Capacitated Facility Location Problem (CFLP) by [Sri95].

2.2 Multi-echelon models

The facility location problems described in the previous section consider only transportation from the distribution centers to the customers. As there is only one level of distribution centers, the network is called a single-echelon network. Such an approach excludes transportation costs from the plants to the distribution centers. It also does not model the frequent network setup in which a large central distribution center supplies smaller regional distribution centers. A network with only two levels, distribution centers and customers, is called a single-echelon network and one with more than two is called a multi-echelon network. [ŞS07] state that ”hierarchical systems are complex systems where an effective coordination of services provided at different levels requires integration in the spatial organization of facilities”.

A precise nomenclature for multi-echelon networks is provided by [ŞS07]. They call a network a *k-echelon network* if it contains k levels of facilities, for which opening and closing decisions are made. According to this definition, the customers are on *level 0*. The customer locations are exempt from the opening and closing decisions. The lowest level of facilities inside the decision scope of the model are those on *level 1*. Consequently, the locations on the highest level are on *level k*.

Multi-echelon systems appear frequently in other areas such as health care ([RS00]), education and waste management ([BDS98]). To illustrate the applicability of multi-echelon networks, the situation in a typical health care system is considered. A patient initially goes to see his doctor, whose office is close to the patient's home. Since many medical devices are not available at the doctor's office, the patient must go to the hospital for procedures involving such devices. If the patient's disease is very rare he must go to one of just a few specialized clinics, because equipment and capable doctors to treat it are only available there. In the example the doctors would be on level one, the hospitals on level two and the specialized clinics on level three.

Multi-echelon facility location problems can be classified according to four characteristics introduced by [ŞS07].

- **Flow patterns**
  A supply chain model is said to be *single-flow*, if products can only be shipped to locations on the next lower level. If shipments can be made to a location on a level below that, then the model is a *multi-flow* model. As an example of a multi-flow model imagine a central warehouse, which replenishes the regional distribution centers and simultaneously acts as a regional one, shipping directly

to customers in the vicinity of the warehouse. In a single-flow model all goods must flow from the central warehouse to the regional one and can only be sent to the customer from there.

- **Service varieties**
  ”In a nested hierarchy, a higher-level facility provides all the services provided by a lower level facility and at least one additional service. In a non-nested hierarchy, facilities at each level offer different services” ([ŞS07]). Thus, a system, in which customers could source from any distribution center or directly from a factory, is a nested system. According to this definition the health care example is a nested network.

- **Spatial characteristics**
  ”In a *coherent* system all demand sites that are assigned to a particular lower-level facility are assigned to one and the same higher-level facility” ([ŞS07]). Thus, in a coherent system, if customers A and B place their orders at warehouse W, then W receives the quantities necessary to satisfy these orders from exactly one factory F. In an incoherent system W receives the quantities it ships to A from a factory F_1 and the quantities it ships to B from factory F_2.

- **Objective**
  The first distinction, which can be made, is whether a model has a single objective function or multiple ones. For the more frequent case of a single objective function several approaches have been presented in the literature. The three most frequent ones are the *median*, *coverage* and *fixed charge* objective functions. In a model with a median objective function the aim is to minimize the total transportation costs. An example of such a model is the Steiner-Weber model. In a coverage model the goal is ”to minimize the total number of facilities needed for covering all customers”. A fixed charge model aims at minimizing the total network cost, including transportation and facility costs under the constraint of satisfying all demands. In a survey of the literature on facility location problems published in 2009 [MNG09] examined, which type of objective function was most frequently used. They found that 75% of the studied papers consider only cost, 16% consider profit and 9% deal with multi-objective problems. These multi-objective problems can include service considerations, which have risen in importance due to more demanding customer requirements.

2.3 Models with inventory integration

The three key elements of a supply chain are locations, transportation and inventories and according to [OCD08] they are ”highly related”. Consequently, a holistic approach to Supply Chain Network Design should consider all three elements.

However, inventory considerations had long been excluded from supply chain design problems. A major reason is that inventory levels are non-linear in the quantity flowing through a location. The higher the throughput of a product at a location, the lower the ratio of inventory to throughput. A more detailed discussion of the theory behind inventory levels will be provided in chapter 3.

Inventory considerations were, for the first time, included in the analysis of a facility location problem by [EM00] in their paper ”The interaction of location and inventory in designing distribution systems”. They considered continuous demand over a square grid.

Further developments were driven by [SCD03], [MSQ07], [OCD08] and [STS05]. The model studied in these papers was motivated by the problem of storage of donated blood platelets. A central supplier provided the platelets for a large group of hospitals. At each individual hospital the demand for blood platelets fluctuates significantly. This mandates large safety stocks to guarantee a high service level. Due to the short shelf life of five days, either a lot of blood platelets had to be thrown away on a frequent basis, if they had not been needed, or the blood had to be procured via express delivery at higher cost and with significant delays. The combination of expensive platelets and a high rate of obsolescence resulted in elevated inventory costs. To guarantee a good service level at low cost the idea was formulated to make some hospitals act as local distribution centers. At these distribution center locations sufficient quantities of platelets should be stored for the hospitals in their vicinity. Whenever a hospital needs platelets, then the distribution center can supply them quickly.

Due to the risk pooling effects of centralized inventory and the shorter lead times, the system wide safety stocks required to guarantee a high service level fell substantially. In addition, the frequency of express shipments from the manufacturing location also decreased, because a larger amount was available at a nearby location.

Various algorithms have been proposed to solve the described problem. [SCD03] used a set-covering algorithm, whereas [OCD08] applied Lagrangean relaxation.

Finding optimal solutions for facility location problems with inventory is hard, due to the introduction of non-linear terms. This has lead researchers to study heuristics for the problem. Two heuristic approaches, Tabu search and particle swarm optimization were used by [CG+12]. Their work considers fixed location costs as well as transportation, inventory and order costs. They place a stochastic constraint on the capacity of the warehouses. However, they consider only a single product.

Genetic algorithms have also been used to solve facility location models in forward logistics ([ABX14] and [Tiw+10]) and reverse logistics ([MJKSK06]).

Both [ABX14] and [Tiw+10] have included multiple products and direct shipping, allowing factories to send products directly to customers. Moreover, [Tiw+10] included non-linear transportation costs to model economies of scale and described a multi-echelon network. Their algorithm incorporates elements of Taguchi Method and Artificial Immune Systems. While safety stocks do not play a role in reverse logistics, [MJKSK06] looks at the average inventory in the system which depends on the non-linear time between collections.

In the next chapter, the theoretical background behind inventory management and the calculation of adequate stock levels will be presented.

Chapter 3

Inventory theory

For several years companies have tried to reduce their inventory levels in order to free up capital and reduce the costs incurred for holding inventory. [CBL96] state that a large element of those inventory carrying costs are capital costs. They can be understood as interest or opportunity costs. Variable costs for storage space and insurance also are an element of inventory carrying costs, as are ”risk costs” such as product obsolescence and declines in value.

However, as [CBL96] observe, carrying physical distribution inventories has several benefits. Inventory allows savings in production and transportation due to economies of scale. Moreover, it allows companies to guarantee better service to their customers and to balance seasonal demand peaks by producing goods ahead of time.

In the next section the motivation for including inventory considerations in the model will be discussed. An overview of the different approaches to calculate stock levels will be given in section 3.2. Finally, these results will be summarized and the formulation used for the remainder of this paper will be presented in section 3.3.

3.1 Motivation of inventory consideration

As shown in section 2.3, inventory had first been integrated into facility location problems in the early 2000s, by [SCD03] and others. Inventory introduces a new driver towards fewer distribution centers, because inventory costs decrease, as more customers are pooled together. Consider n independent[3] random variables X_i with $\mathbb{E}(X_i) \geq 0$, which represent demand at location i. Let random variable $Y = X_1 + ... + X_n$ be the sum of all demands. The standard deviation σ and the expected demand μ of the

[3]Inventory reduction due to pooling can also be observed, if the random demands are weakly correlated

random variable Y are given by

$$\sigma = \sqrt{\sum_{i=1}^{n} Var(X_i)}$$

$$\mu = \sum_{i=1}^{n} \mathbb{E}(X_i)$$

The coefficient of variation of the pooled demand $\frac{\sigma}{\mu}$ tends to decrease, as the number of aggregated locations n increases.

The impact of this will be explained using the newsboy model. In this model a vendor has to decide how many newspapers to order for the following day. Assume a population of $n = 10$ people. For each person i the binary independent random variable $X_i = 1$ if the person buys a newspaper on a day and $X_i = 0$ if not. We have $P(X_i = 1) = \frac{2}{3}$. Let Y_k be the sum of k random variables. The cost of not having sold a newspaper $c_o = p$ is equal to the purchasing price p. In contrast, the cost of not being able to satisfy a demand $c_u = s - p$ is the difference between the sales price s and the purchasing price p. It is the lost profit which the newsboy could have realized, had he had another newspaper available. Let us now compare two distinct setups. In the first setup two newsboys Tom and Jerry work independently. Tom sells to persons 1 to 5 and Jerry sells to persons 6 to 10. They both have the same cost structure with $c_o = 1$ and $c_u = 2.5$. Since the random variables are independent and identically distributed the total cost $Z(S)$ given an order quantity S and random demand Y_k with $k = 5$ is

$$Z(S) = c_o \sum_{n=1}^{S} (S - n)P(Y_k = n) + c_u \sum_{n=S}^{\infty} (n - S)P(Y_k = n)$$

In the following table one can see that costs of each newsboy are minimal, if he purchases $S = 4$ newspapers. He would then incur a cost of 1.13. Consequently, the combined cost of both newsboys is $2 \cdot 1.13 = 2.26$.

Order quantity S	Overage cost c_o	Underage cost c_u	Total cost
0	0	8.33	8.33
1	0	5.84	5.84
2	0.04	3.57	3.61
3	0.26	1.48	1.74
4	0.80	0.33	1.13
5	1.67	0	1.67

Compare this with the case in which Tom and Jerry pool their demands. Using the same analysis as in the previous table, but with $k = 10$, the optimal combined order quantity remains at $S = 8$. But, due to the demand pooling, the total cost incurred decreases by 22% from 2.26 to 1.76. The reason for this improvement lies in the fact that sometimes, when all 5 of Tom's customers want to buy a newspaper, only 3 or less of Jerry's customers will want to buy one. In this case Jerry would have one newspaper left, which he could loan out to Tom, and it is more likely that all $S = 8$ newspapers are sold on a given day and less likely that a demand has not been satisfied.

According to [Coy09], "inventory carrying costs are on average 33% of total logistics costs for organizations". As a result, the drastic cost saving potential of pooled demands ought to be included in a holistic strategic supply chain study.

3.2 Different approaches to inventory calculation

Multiple approaches to inventory management exist. Frequently, a distinction is drawn between *pull* and *push* policies. [Bow13] notes that pull policies "use customer demand to pull product through the distribution channel", while push policies use "a planning approach that proactively allocates or deploys inventory on the basis of forecasted demand and product availability".

Forecasting relies on statistical methods. Incorporating such methods into a facility location problem is not suitable, because of the added complexity. Since facility location problems use a priori demand estimations, a pull policy will be employed. Within pull policies one distinguishes between continuous and periodic review policies. Continuous review policies track the inventory level at any moment. [Bow13] contrast this with periodic review policies, under which a statement about the inventory level can only be made at the beginning and the end of a review period. For example at the end of the week. A pull policy requires a rule, which dictates, when an item should be replenished and how many units should be ordered. A new order is placed whenever the onhand stock drops below a defined quantity, the *reorder point*. At this time a policy specific quantity, the reorder quantity, is ordered.

[Tho06] states that the continuous review model is more complex than a periodic review model, because reorder point and reorder quantity are optimized simultaneously. In order to reduce complexity with regards to inventory, a periodic review policy will be used in the remainder of this thesis.

In inventory theory two types of stock, *cycle stock* and *safety stock*, are distinguished according to [Tho06].

3.2.1 Cycle stock

Cycle stock is the part of the inventory which covers the expected demand during the review period. In the cycle stock calculation a fixed average demand per review period is assumed. The typical inventory pattern underlying the cycle stocks can be seen in figure 3.1.

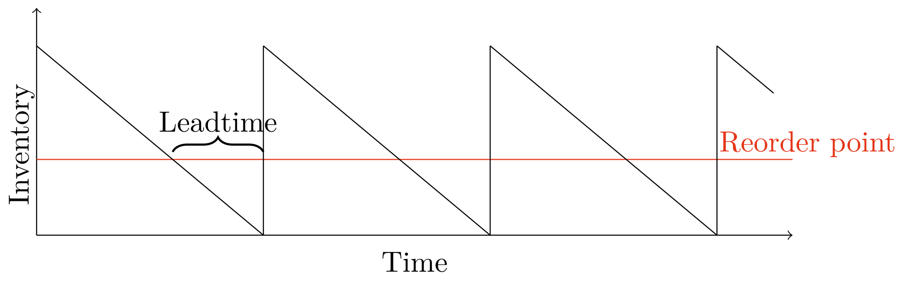


Figure 3.1: Order is triggered any time the inventory falls below the reorder point denoted by the red line. The inventory arrives after the defined lead time.

For ease of notation and due to practical applicability the review period is assumed to be one week. Cycle stock depends on the average weekly demand μ. Given a reorder interval ri, a company orders a quantity $Q = ri \cdot \mu$ every ri weeks. Assuming deterministic demand and allowing partial units, the onhand inventory $i(t)$ at a time $t \in [0, ri]$ is $i(t) = Q - t\mu$. On average the cycle stock level during the reorder interval is $CS = \frac{\mu \cdot ri}{2}$.

The cycle stock depends linearly on the demand and on the actual reorder interval. However, the reorder interval is not necessarily a constant. Frequently, a desired value for the reorder interval cannot be met, because the supplier demands a Minimum Order Quantity MOQ. It can happen, that the demand during the reorder interval $ri \cdot \mu$ is lower than the minimum order quantity. In this case the order interval must be prolonged, because otherwise large amounts of inventory would accumulate in the warehouse. Thus, the reorder interval must be at least $ri \geq \frac{MOQ}{\mu}$. Consequently, the cycle stock held at a location is equal to

$$\begin{aligned} CS &= \frac{\mu}{2} \max\{ri, \ \frac{MOQ}{\mu}\} \\ &= \frac{1}{2} \max\{ri \cdot \mu, \ MOQ\} \end{aligned} \tag{3.1}$$

3.2.2 Safety stock

While cycle stock covers the anticipated regular demands, safety stock on the other hand is kept to deal with unpredictable demand fluctuations. Since companies want to guarantee their customers a high service level, they keep safety stock to guard against above average demand and the resulting stock outs. The service level can be defined as the percentage of demand satisfied. However, this definition is not a precise one and allows for various interpretations. Two of the most common interpretations of service level are

- α **- Service Level** representing the probability of being able to fill all customer demands in full during a review period.

$$\alpha = P(\text{demand during lead time } \leq \text{ onhand inventory at start of period})$$

- β **- Service Level** describing the expected percentage of demanded units, which can be supplied without delay. When X is a random variable describing the expected backorders at the end of an interval and μ the expected demand during the interval.

$$\beta = 1 - \frac{\mathbb{E}[X]}{\mu}$$

Very rarely the γ - service level is used. It takes into account the percentage of demanded units which are supplied with a delay while simultaneously measuring the length of the delay. It is not used frequently, because estimating the length of the delay is not a trivial endeavor.

Since the calculation of the β - service level under a periodic review policy is difficult according to [Tho06], the α - service level will be used. A target stock level is defined which should only be touched when demand is higher than expected during the reorder interval. It is assumed that demand is normally distributed with variance σ^2 during the review period r. The lead time [4] lt in multiples of the review period r is known and deterministic. Thus, the variance during review period and lead time $\sigma^2_{lt+1} = \sigma^2 \ (lt+1)$. Given a service level α

$$\begin{aligned} SS &= F^{-1}(\alpha) \ \sigma_{lt+1} \\ &= z \ \sigma_{lt+1} \\ &= z \ \sqrt{\sigma^2}\sqrt{lt+1} \end{aligned} \tag{3.2}$$

3.3 Conclusion

Due to the arguments in section 3.1 inventory will be considered in the model presented in chapter 5. Both, cycle stock and safety stock, will include a non-linear element. The non-linearity in the cycle stock is introduced by the maximum function. In the safety stock formula the non-linearity stems from the two square root term.s The safety stock non-linearity is more severe since the maximum function is linear once $Q = \mu \cdot ri \geq MOQ$.

To summarize the results of the previous section, the inventory levels used in the model will be based on a pull policy with periodic review. Cycle stock levels are given

[4]Lead time describes the time between placing an order and receiving the product

by the equation

$$CS = \frac{1}{2}\max\{ri \cdot \mu,\ MOQ\} \tag{3.3}$$

For safety stock calculations the α service level is used with $z = F^{-1}(\alpha)$. The formula for the safety stock is given by

$$SS = z\ \sqrt{\sigma^2}\sqrt{lt + 1} \tag{3.4}$$

Chapter 4

Problem description

This thesis attempts to answer the question how the distribution of a company's products to its customers can be organized in such a way, as to minimize costs while simultaneously satisfying the customers' service requirements. In real-life networks this question is complicated, because of varying lead time requirements, a complex network structure and distinct products manufactured at many different factories.

In section 4.1 the questions which the model shall answer are posed. Then, the assumptions of the studied network and their implications on the optimization are discussed in section 4.2. Finally, in section 4.3 the multi-echelon model is classified according to the characteristics defined by [ŞS07]

4.1 Answers generated by the model

As seen in chapter 2, the facility location problem aims to select the optimal set of locations at which to operate distribution centers and to find the optimal product flows between them. This thesis extends the typical facility location problem in two directions. On the one hand, two echelons of distribution centers are considered. This means, that one distribution center can be chosen from a set of central distribution centers. In addition, there is a larger set of regional distribution centers, from which a subset is selected. On the other hand, a large set of distinct products is considered. Considering multiple products adds complexity, since the paths of two products from factory F to customer C can be different. While one product may be distributed from factory F to customer C through the RDC, another product could be stored centrally at the CDC and at the RDC.

Since production locations are highly inflexible, it will be assumed that production locations are fixed. Moreover, average weekly demand and variance at each customer are also known. The sets of potential CDC and RDC locations are also given. All costs are linear with respect to their quantities. This includes inventory costs, which are linear in the inventory held. The inventory non-linearities appear, because the onhand inventory is non-linear in the demand and variance at the locations.

The questions which this thesis attempts to answer are

1. Should there be only one level of distribution centers in the network, i.e. single-echelon, or should there be an additional central distribution center, and thus a two-echelon network?
2. If there is a central distribution center, where should it be located?
3. Which of the possible regional distribution center locations should be used?
4. How much of each product is sent from one location to another?
5. From where are customers supplied each product?
6. How high are product cycle stock and safety stock at each open location?

Questions 1 through 5 are strategic, as any decision taken in that regard is not easily reversible. While decisions about inventory levels are frequently classified as tactical, they still heavily influence the strategic decisions.

Now that the questions are posed, it is important to understand which assumptions underlie the model.

4.2 Discussion of assumptions about the network

Networks are intrinsically complex and contain many customized elements, which cannot be incorporated in a high level strategic model. Thus, in order to create a solvable model, assumptions must be made about the locations, products, product flows and cost elements.

Locations

Since the focus of the model shall be on the distribution of finished goods from factories to customers, these two sets of locations must be represented in the model. The results of the model only affect the intermediate distribution center levels directly. While there are networks with more than two levels of distribution centers, they are relatively rare and more complex. Therefore, only networks with two layers of distribution centers can be tackled within the model. One of the critical decisions in supply chain design is, whether a two-echelon network with a central distribution center and a few RDCs is better than a single-echelon network with no CDC and many RDCs.

Inventory at the factories is difficult to model, because detailed information about production schedules and procurement processes is required to determine a lead time. Therefore, the inventory at source locations is not considered. In summary, the following assumptions with regard to the locations are formulated.

- There are 4 different sets of locations: Customers, regional distribution centers (RDCs), central distribution centers (CDCs) and sources
- Factories and customers are fixed and cannot be changed
- Either one CDC is open or none
- Multiple RDCs can be open
- Inventory is only held at the RDC and CDC locations

Products

Most companies produce a wide variety of products, ranging from a handful of homogeneous ones in the case of, for example, Coca Cola, to more than a hundred thousand distinct products with very heterogeneus properties produced by car companies. Many companies produce each product at only one location.

For each product and each customer the demand fluctuates over time in a stochastic manner. Demand, thus, can be understood as a random variable with an expected weekly demand and a weekly variance. Using these two values, inventories can be estimated according to the formulas discussed in chapter 3.

In summary, the following assumptions can be formulated with regard to the products.

- There are multiple products
- Each product is produced at exactly one source location
- Demand of each product at a customer location is stochastic with known weekly demand and variance

Product flow

As established previously, all products are produced at a source location and consumed at various customer location. Thus, any flow must start in a source location and end at a customer location. No reverse flows, e.g. from an RDC to the CDC, or flows between two locations on the same level, e.g. two RDCs, are allowed. While such flows can occur in practical networks, they generally are not an expected part of of the regular setup. Most of the time such flows occur in order to cope with unexpected events, such as high one-time demands. These events are beyond the scope of a strategic analysis and are consequently not included.

Since the methodology lined out in this thesis is demand-riven, all customer demands must be met. They can receive their goods from any location. Thus, if a customer would be willing to wait for a very long time and would order a very large quantity, he could receive the product directly from the factory. The more important case is, that the CDC could effectively act as both, a central distribution center replenishing the CDCs and simultaneously as a regional distribution center satisfying customer demands.

Given that most customers order amounts of a similar order of magnitude, when they order a product, they will receive a product from one location. While in reality there may be exceptions, it is difficult to identify them and deal with them accurately within the framework of a strategic model. Therefore, such cases are not included.

In summary the following assumptions hold about product flows.

- The path of every product flow must start at a source and end at a customer.
- All demands must be satisfied.
- The demand of a product at a location is satisfied by exactly one location.
- Product flow can only occur downstream, i.e. Source $\rightarrow$ CDC $\rightarrow$ RDC $\rightarrow$ Customer. On this path the stops CDC and RDC can be omitted.
- As a corollary of the above statement, a customer's demand can be satisfied by a flow from any system location, i.e. source, CDC or RDC.

Cost calculation

The warehouse operating cost is a fixed charge which must be paid if a warehouse is used. Variable handling costs at a distribution center are based on the number of lines, which are handled at a distribution center. The handling costs are linear in the amount of lines. Inventory costs are linear in the amount of inventory held. It is important to note that the non-linearity of inventory costs is rooted in the fact that the amount of inventory is non-linear in the demand and the variance. The non-linearities complicate the problem in a substantial manner, but help to make it more precise.

The linearity of transportation costs is a more difficult assumption. In reality, costs to ship a good from A to B are not linear in weight and distance, because economies of scale influence the prices. But, for a route from A to B a good estimate for the average shipment weight can be found. Based on this average it is possible to estimate a valid per-kilogram price for this route, assuming that shipment quantity remains relatively stable.

In summary, the following assumptions hold about the calculation of costs.

- Transportation cost, warehouse operating cost, product handling cost and inventory cost are all linear with respect to their cost driver
- Orders at a location must be larger than the minimum order quantity (MOQ)
- Inventory at a distribution center follows a non-linear function
- A periodic review inventory policy is used

4.3 Classification of the model

In the following the model is compared to other models, which have been presented in the literature. This is done using the characteristics formulated by [ŞS07] to classify multi-echelon facility location problems. According to these characteristics, which were discussed in section 2.2, the problem can be classified as a coherent, nested, multi-flow fixed-charge model.

- It is coherent, because any lower level facility will only be allowed to source a specific product from one higher level location.
- Since locations on a level may ship to any location on a lower level, the model is nested.
- Since direct shipments between two non-adjacent levels of the supply chain will be allowed, it is a multi-flow problem. This means, that a factory can ship directly to an RDC in its vicinity.
- A fixed charge model is one, in which the entire network costs are to be optimized. This thesis attempts to include a large portion of the actual costs, including penalties for not meeting a customer's lead time requirements.

Chapter 5

Mathematical modeling

In this chapter the mathematical model will be laid out. The goal of the model is to describe a company's distribution supply chain according to the assumptions laid out in chapter 4. As a result of this chapter a mixed integer non-linear program will be formulated describing a multi-echelon multi-product distribution supply chain including inventory costs.

In the next section the input parameters will be defined. The cost elements included in the objective function will be explained in section 5.2. Finally, the mixed integer non-linear program will be formulated in section 5.3.

5.1 Definition of parameters and variables

In this section the input parameters will be defined. At first the sets used to describe a distribution supply chain will be specified. In the following, the customer, system location and relation parameters will be defined, before the product parameters will be laid out.

5.1.1 Sets used to describe the network

The basic parts which make up a distribution network are locations, relations and products. The different elements are referenced using the definitions below. There are multiple sets of locations which appear in the network.

- N is the set of all locations
- $D \subset N$ is the set of all customer demand locations
- $S \subset N$ is the set of all source locations
- $W \subset N$ is the set of all distribution center locations. In more detail the set W_R refers to all RDCs and W_C refers to all CDCs.
- $V = (W \cup S) \subset N$ is the set of all system locations

- For each node $i \in N$ the set $S^{-}(i)$ is the set of all nodes $j \in N$ with arc $(j, i) \in E$, i.e. on a higher level
- For each node $i \in N$ the set $S^{+}(i)$ is the set of all nodes $j \in N$ with arc $(i, j) \in E$, i.e. on a lower level

Based on the set of locations N we define the set of arcs and also the set of products

- $E \subset N \times N$ is the set of all arcs (j, i) which are allowed to be used in the network
- P is the set of all products
- P_l is the set of all products produced at location $l \in S$

5.1.2 Customer input parameters

For each location $i \in D$ and each product $p \in P$ there are several input parameters. These are information about demand quantity, variance and number of orders. The demand quantities determine how much of each product flows through the network. The variances influence the safety stock calculations and the number of orders impacts the outbound handling costs at the distribution centers.

- μ_i^p is the average weekly demand of product $p \in P$ at customer location $i \in D$
- $(\sigma_i^p)^2$ the weekly variance of product $p \in P$ at customer location $i \in D$
- τ_i^p is the average weekly number of lines, i.e. how many times product $p \in P$ is ordered at customer location $d \in D$.

In addition, there is information about the customers' required lead time. This information can be either on a customer level or on a product - customer level. If a customer lead time requirement is defined, then the weighted average of all products' lead times is taken and compared to the requirement. If the lead time requirement is formulated on the product - customer level, then the lead time of each individual item is compared to its specific requirement.

- l_i is the time in which an order at location $i \in N$ must on (weighted) average be fulfilled.
- s_i is the lead time penalty costs for missing customer lead time requirement at location $i \in D$ by one week.
- l_i^p is the time in which an order of product $p \in P$ at location $i \in N$ must be fulfilled.
- s_i^p is the lead time penalty costs for missing customer lead time requirement of product $p \in P$ at location $i \in D$ by one week.

5.1.3 Input parameters for system locations

At the system locations there are two types of input parameters. On the one hand are the costs, such as the annual operating cost, and on the other hand are parameters describing non-monetary aspects of the location. These aspects include, for example, minimum order quantities.

Cost elements at system locations

There are fixed costs at a system location which are incurred if the location is opened. The variable costs at the location are incurred each time a product is received at a location (inbound) or sent from a location (outbound).

- fwc_j is the annual cost of opening and operating location $j \in V$
- vwc_j^p is the per unit cost of handling one line of product $p \in P$ at location $j \in V$

Non-cost parameters at system locations

With regard to inventory there are additional parameters which must be provided at the location level, such as the desired reorder interval of a product and the review period.

- τ_i^p is the average weekly number of lines, i.e. how many times product $p \in P$ is ordered at system location $d \in V$.
- RI_j^p is the desired reorder interval of product $p \in P$ at location $j \in W$.
- MOQ_k^p is the minimum order quantity when ordering product $p \in P$ from location $k \in W_C \cup S$.
- $Q_j^p(\mu_j) = max\{RI_j\ \mu_j, MOQ_k\}$ is the actual order quantity of product $p \in P$ at location $j \in W$, when ordering from location $k \in W_C \cup S$. The order quantity depends on the decision variable μ_j, which is the demand assigned to the location j.
- r_j is the review period at a location $j \in W$. It is the time between inventory reviews. At these reviews an order is placed, if the on-hand inventory is below a threshold. It impacts the amount of safety stock which must be kept.

5.1.4 Input parameters for relations

Relations are a major cost driver and also heavily affect the lead time requirements. The annual transportation cost from a system location to a customer can be calculated a priori, because the quantity and frequency are known. This is not possible for relations between two system locations, for which a per unit cost must be calculated. The replenishment lead time between two locations can be computed for any pair of locations and consists of the transportation time and the time required to prepare the product for shipping.

- c^p_{ji} denotes the transportation cost of supplying the entire demand of product $p \in P$ at a customer $i \in D$ from a system location $j \in V$.
- c^p_{kj} denotes the transportation cost of sending one unit of product $p \in P$ from one system location $k \in V$ to another system location $j \in V$.
- rlt_{ji} is the replenishment lead time of sending a product from location $j \in V$ to another location $i \in N$.

5.1.5 Input parameters at product level

Each product has a distinct inventory holding cost. Frequently inventory holding cost is given as a percentage of product cost and thus a product's actual inventory holding cost ihc^p is the product of that percentage and the product cost. In addition, the required service level α_p of products can be different. Therefore, the variable z in equation 3.4 must be replaced by a product specific variable $z_p = F^{-1}(\alpha_p)$.

- ihc^p is the annual inventory holding cost of an item $p \in P$
- z_p is the inverse of the standard normal function for a product $p \in P$ with service level α_p.

5.2 Objective Function

The objective function represents the total cost of the distribution network TNC. The total network cost is the sum of transportation costs TC, inventory costs IC, warehouse operating costs WC and penalty costs PC. The goal is to minimize the Total Network Costs

$$TNC = TC + IC + TWC + PC \tag{5.1}$$

5.2.1 Transportation Cost

Transportation costs can be split into two parts. The customers must be supplied and their demands, both in quantity and in frequency, are independent of the locations of the distribution centers. Thus, it is possible to determine the hypothetical costs to supply a customer from any location a priori. The quantities between own locations, on the other hand, are not known, because they depend on the assignment of customer demands. Therefore, these two types of transportation cost, the cost to ship to customer locations $TC_{Customer}$ and the cost to ship to own locations TC_{System}, are determined independently.

$$TC = TC_{Customer} + TC_{System} \tag{5.2}$$

Transportation Cost to customers

For the shipments to customers an estimate of the total annual cost of supplying them with a product can be found, because the annual quantity $\bar{\mu}_i^p$ and the number of lines from a source $j \in V$ to a customer $i \in D$ are known. This allows the calculation of annual transportation costs to supply all customers.

$$TC_{Customer} = \sum_{p \in P} \sum_{i \in D, j \in V} y_{ji}^p c_{ji}^p \tag{5.3}$$

Transportation Cost in the system

For the transportation cost in the system it is assumed that shipments to the distribution centers are of uniform "size". Based on this assumed size a rate per unit can be calculated.

$$TC_{System} = \sum_{p \in P} \sum_{i,j \in V} y_{ji}^p \mu_j^p c_{ji}^p \tag{5.4}$$

5.2.2 Warehouse Costs

The total warehouse costs WC are the sum of the fixed and variable costs of open warehouses.

$$WC = FWC + VWC \tag{5.5}$$

Fixed Warehouse Costs (FWC)

Fixed Warehouse Costs (FWC) are incurred once the decision to open a distribution center is made. They include opening costs, e.g. construction and relocation, and fixed operating costs such as administrative overhead. At each distribution center location $j \in W$ a cost fwc_j is incurred, if location j is opened.

$$FWC = \sum_{j \in W} x_j fwc_j \tag{5.6}$$

Variable Warehouse Costs (VWC)

Variable Warehouse Costs (VWC) are linear in the number of outbound lines τ_i^p leaving the distribution center. They represent handling costs of outbound shipping and are dependent on the product $p \in P$. The cost of handling one line of product p is vwc_j^p, $\forall j \in V, p \in P$.

$$VWC = \sum_{j \in V,\ i \in S_j^+} \sum_{p \in P} y_{ji}^p \tau_i^p vwc_j^p \tag{5.7}$$

5.2.3 Inventory Cost

The inventory costs IC include all costs of holding a specific amount of inventory of a product $p \in P$.

$$IC = \sum_{p \in P} ihc^p \sum_{j \in W} (SS_j^p + CS_j^p) \tag{5.8}$$

The main cost driver is the stock level which is the sum of safety stock and cycle stock across all locations in the network.

Cycle stock

In equation (3.3) the used cycle stock formulation was presented.

$$CS = \frac{1}{2} \max\{ri \cdot \mu, MOQ\} \tag{5.9}$$

For a location $j \in W$ the reorder interval ri_j^p is given as an input parameter. The demand $\mu_j^p = \sum_{i \in S_j^+} y_{ji}^p \mu_i^p$ is the sum of all demands assigned to location j. The minimum order quantity $MOQ_j^p = \sum_{k \in S_j^+} y_{kj}^p MOQ_k^p$ depends on the location k from which the location j is supplied.

Inserting this in equation (3.3) for a specific location $j \in W$ and a product $p \in P$ we get

$$CS_j^p = \frac{1}{2}\max\{ri_j \sum_{i\in S_j^+} y_{ji}^p \mu_i^p \ , \ \sum_{k\in S_j^-} y_{kj}^p MOQ_k^p\} \tag{5.10}$$

Safety Stock

In equation (3.4) the used safety stock formulation was presented.

$$SS = z\ \sqrt{\sigma^2}\sqrt{lt + r} \tag{5.11}$$

At a location $j \in W$ and for a product $p \in P$ the variables z^p and r_j are input parameters for the product or location. The variance $(\sigma_j^p)^2 = \sum_{i\in S_j^+} y_{ji}^p(\sigma_i^p)^2$ is the sum of all the variances of products which source product p from location j. The lead time lt_j^p depends on the location $k \in V$ from which location j sources. Therefore $lt_j^p = \sum\limits_{k\in S_j^+} y_{kj}^p rlt_{kj}$.

Inserting this in equation 3.4 for a specific location $j \in W$ and a product $p \in P$ we get

$$SS_j^p = z^p \sqrt{\sum_{i\in S_j^+} y_{ji}^p(\sigma_i^p)^2}\sqrt{\sum_{k\in S_j^-} y_{kj}^p rlt_{kj} + r_j} \tag{5.12}$$

5.2.4 Lead time penalty cost

Since a distribution supply chain network has the two objectives of low cost on the one hand and high customer service on the other it is important to penalize not meeting customer lead time targets. To do this, a lead time penalty cost PC is assessed for the average weekly delay multiplied with the number of units. Such costs can represent both contractual penalty cost and less quantifiable costs representing loss of future sales.

For each product p at each customer location i a lead time rlt_{ji}^p is given. This lead time must then be compared to the lead time requirement. Lead time requirements can be defined for each product at a customer location, but also across the entire set of products the customer demands. Only one of the two approaches can be used.

$$PC = \begin{cases} PC_{Product} & \text{for product-customer lead time requirements} \\ PC_{Customer} & \text{for aggregated customer lead time requirements} \end{cases} \tag{5.13}$$

Lead time requirements on product level

In the case of specific lead time requirements l_i^p for each product at a customer location, the lead time violation is given by

$$f_i^p = \max\{\sum_{j \in S_i^-} y_{ji}^p rlt_{ji}^p - l_i^p, 0\} \tag{5.14}$$

Whenever the lead time is higher than the requirement, there is a violation and a penalty $f_i^p \; s_i^p$ is assessed. When using product-customer lead time requirements the total penalty cost is given by

$$PC_{Product} = \sum_{i \in D} \sum_{p \in P} f_i^p \; s_i^p \tag{5.15}$$

Lead time requirements on customer level

In the case of average lead time requirements l_j at each locations, the lead time violation is computed as the average over all products. Therefore one calculates the lead time violation term

$$f_i = \max\{\frac{1}{\sum_{p \in P} \mu_i^p} \sum_{p \in P} \mu_i (\sum_{j \in S_i^-} y_{ji}^p rlt_{ji}^p - l_i), 0\} \tag{5.16}$$

Thus, in the case of using customer lead time requirements the penalty cost at a location i is $f_i s_i$ and the total penalty cost is given by

$$PC_{Customer} = \sum_{i \in D} f_i \; s_i \tag{5.17}$$

5.3 Mathematical Model

In this section a mixed integer non-linear optimization formulation will be laid out. All the input parameters described in section 5.1, are given. The goal of the optimization is to select a subset of CDC and RDC nodes on the network, and to choose flows through the induced network, so as to minimize the total network costs defined in (5.1). No flows may go into or out of a node, which is not part of the selected subset of distribution centers. The inbound flow of any product into any node, which represents a customer, must be equal to the demand of this customer. Moreover, the sum of inbound flows into any distribution center nodes must be equal to the outbound flows from this distribution center.

The decision variables required for the formulation are defined in 5.3.1 and the complete model is presented in 5.3.2

5.3.1 Decision Variables

There are several types of decision variables included in the model. Binary variables are used to describe whether an RDC or CDC location is opened. The decision, from where a location sources its product is also described by binary decision variables. In addition, the lead time violation, either on the product or on the customer level, are given by a non-negative decision variable which depends on the customer's requirements and the actual lead time from the source.

The demand and variance at system locations are given as the sum of the demands and variances of locations which source their products from them. It is important to keep in mind that the number of inbound lines at system locations is an input parameter (see 5.1.5).

- Binary variables $x_j \in \{0, 1\}\ \forall j \in V$ with

$$x_j = \begin{cases} 1 & \text{if location } j \text{ is open} \\ 0 & \text{otherwise} \end{cases}$$

- Binary variables $y_{ji}^p \in \{0, 1\}\ \forall i, j \in N$ and $\forall p \in P$ with

$$y_{ji}^p = \begin{cases} 1 & \text{if location } i \text{ sources product } p \text{ from location } j \\ 0 & \text{otherwise} \end{cases}$$

- $f_i \geq 0$ is the average time by which the lead time l_i at location $i \in D$ is overshot

- $f_i^p \geq 0$ is the average time by which the lead time l_i^p of product $p \in P$ at location $i \in D$ is overshot

- μ_j^p and $(\sigma_j^p)^2$ with $j \in V$ and $p \in P$ are the demand and variance at a non-customer location and are the sum of the demands of those locations which source from location j.

5.3.2 Mixed integer non linear problem formulation

$$
\begin{aligned}
\min \quad TNC \quad & = TC + WC + IC + PC && && (5.18a)\\
\text{subject to} \quad & && && \\
\sum_{j \in V} y_{ji}^p \quad & = 1 && \forall\, i \in D,\ p \in P && (5.18b)\\
\sum_{i \in D} y_{ji}^p \quad & \leq x_j && \forall\, j \in W,\ p \in P && (5.18c)\\
\mu_j^p \quad & = \sum_{i \in S_j^+} y_{ji}^p \mu_i^p && \forall\, j \in V\ p \in P && (5.18d)\\
(\sigma_j^p)^2 \quad & = \sum_{i \in S_j^+} y_{ji}^p (\sigma_i^p)^2 && \forall j \in W,\ p \in P && (5.18e)\\
\sum_{k \in S_j^-} y_{kj}^p \mu_j^p \quad & = \sum_{i \in S_j^+} y_{ji}^p \mu_i^p && \forall\, j \in W\ p \in P && (5.18f)\\
x_j \quad & \in \{0,1\} && \forall\, j \in W && (5.18g)\\
y_{ji}^p \quad & \in \{0,1\} && \forall\, j \in V,\ i \in S_j^-,\ p \in P && (5.18h)\\
f_i \quad & \geq 0 && \forall\, i \in D && (5.18i)
\end{aligned}
$$

Constraint (5.18b) ensures that all customer demands are satisfied. Constraint (5.18c) states, that demand can only be assigned to a distribution center, if it was opened. The demand μ_j^p and variance $(\sigma_j^p)^2$ at a location $j \in V$ are set in the constraints (5.18d) and (5.18e). Constraint (5.18f) in conjunction with (5.18d) ensure that an outbound flow is met with an equivalent inbound flow at a location. The next two constraints (5.18g) and (5.18h) are simple binary constraints.

To allow for the inclusion of lead time penalty cost the variables f_i representing the lead time penalty factor are set to a non-negative number by constraint (5.18i). With the help of two additional constraints lead time constraints can be defined. In conjunction with constraint (5.18i) the constraints (5.19a) and (5.19b) provide an equivalent formulation of equation (5.14) and equation (5.16) respectively.

$$
\begin{aligned}
f_i^p &\geq \sum_{j \in V} rlt_{ji}^p y_{ji}^p - l_i^p && \forall\, i \in D,\ \forall\, p \in P && (5.19a)\\
f_i &\geq \frac{1}{\sum_{p \in P} \mu_i^p} \sum_{p \in P} \sum_{j \in V} \frac{rlt_{ji}^p y_{ji}^p}{\mu_i^p} - l_i && \forall\, i \in D && (5.19b)
\end{aligned}
$$

Chapter 6

Development of algorithm

The goal of this chapter is to give an overview of the proposed approach to solving the mixed integer non-linear program described in section 5.3. Some of the issues pertaining to the mixed integer program will be discussed in section 6.1, and the reason why linear optimization techniques are not suitable will be addressed. In section 6.2 the motivation for using a heuristic instead of an exact algorithm will be outlined. A two part heuristic approach will then be presented in section 6.3. In that section an overview of the algorithm and a justification for breaking the solution process into two parts will be provided. The approach combines genetic algorithms, described in chapter 7, and the solution of many small and relatively simple independent flow problems, for which a solution procedure will be described in chapter 8.

6.1 Problems of the non-linear optimization formulation

Given, that the presented problem is a non-linear problem, the initial reaction might be, to use a non-linear solver for the problem. However, given the number of binary variables and the general problem size, such an approach would take too long.

The next idea was to use piecewise linear approximation, similar to the one proposed by [Yao+10], who modeled a network with only one echelon of distribution centers. They approximate the square root term over the sum of variances with a piecewise linear function, and then used linear optimization.

Introduction of binary variables due to linear approximation of non-linear terms

Linear approximation of a non-linear function requires that the function $f(x)$ is evaluated at a set of defined points b_i in a set B. For each interval $[b_i, b_{i+1}]$ a linear function $\hat{f}_i(x)$ is defined, with $f(b_i) = \hat{f}_i(b_i)$ and $f(b_{i+1}) = \hat{f}_i(b_{i+1})$. For any value $x \in [b_i, b_{i+1}]$ the value of $f(x)$ is approximated with the value $\hat{f}_i(x)$.

When $|B|$ break points are used to approximate the square root term for one product, then $|B| \cdot |W|$ additional binary variables are introduced into the problem, since the non-linear function must be approximated for all warehouse locations in the set of distribution centers W. In a model with multiple products, i.e. with $|P| > 1$, the number of binary variables required to approximate the square roots would grow by a factor of $|P|$. This results in $|B| \cdot |W| \cdot |P|$ binary variables. Adding such a significant number of binary variables increases the problem complexity. In addition, [Yao+10] only looked at a single-echelon problem and extending the problem to a multi-echelon problem leads to further complications, because it is not possible to only look at the flows across an edge y_{ji}^p.

Effect of replacing edge variables with path variables

When looking at equation (5.18e), it can be seen, that for a CDC location k

$$\begin{aligned}
(\sigma_k^p)^2 &= \sum_{i \in S_k^+} y_{ki}^p (\sigma_i^p)^2 && \forall\, p \in P \\
&= \sum_{i \in S_k^+ \cap D} y_{ki}^p (\sigma_i^p)^2 + \sum_{j \in S_k^+ \cap W} y_{kj}^p (\sigma_j^p)^2 && \forall\, p \in P \\
&= \sum_{i \in S_k^+ \cap D} y_{ki}^p (\sigma_i^p)^2 + \sum_{j \in S_k^+ \cap W} \sum_{i \in S_j^+ \cap D} y_{kj}^p y_{ji}^p (\sigma_i^p)^2 && \forall\, p \in P
\end{aligned}$$

In the second term of the last equation a term of the form $y_{kj}^p y_{ji}^p$ is present. Consequently, the term is no longer linear and instead becomes quadratic. To mitigate this issue and make the problem linear, new binary decision variables need to be introduced. These new decision variables no longer model flows over an edge, but rather along a path through several nodes. These variables would be of the form

$$y_{lkji} = \begin{cases} 1 & \text{if there is a flow from source } l \text{ via CDC } k \text{ and RDC } j \text{ to customer } i \\ 0 & \text{otherwise} \end{cases}$$

Given $|W_C|$ CDCs, $|W_R|$ RDCs and $|D|$ customers, a reformulation replacing edge variables with path variables would result in $|W_C| \cdot |W_R| \cdot |D|$ binary variables per product. The number of source location does not come into play, because it was assumed that each product is only produced at one source location. This number is only a rough estimate, because the flows which do not go through a CDC or RDC are not yet included.

In comparison, to model the same network with edge variables, $|W_C| \cdot |W_R|$ variables are required to describe the flows from the CDCs to the RDCs and $(|W_C| + |W_R|) \cdot |D|$ variables to model the flows from all the DCs to all the customers. Thus, the total amount of variables required would be $|W_C| \cdot |W_R| + (|W_C| + |W_R|) \cdot |D|$.

Since $|D| \gg |W_C|, |W_R|$, the driving factor for the problem size lies in the terms containing $|D|$. Therefore, it is important to look at the behavior of $|W_C| + |W_R|$ compared to $|W_C| \cdot |W_R|$. Assuming a relatively small problem with $|W_C| = 5$ and $|W_R| = 10$, the edge formulation would have approximately $\frac{5 \cdot 10}{5+10} = \frac{10}{3}$ times as many binary variables.

Combined effect of the changes to the formulation

The combined effect of the use of path variables and the piecewise approximation of non-linear elements quickly enlarges the problem by a significant factor. A simplified model has been implemented. In this simplified model flows from the source locations were not possible. Moreover, the objective function only was the sum of the fixed warehousing costs and the safety stock inventory costs, which were assumed to be equal at all locations. In this case the optimal solution is to open only the location with the lowest costs. The problem included 100 customers, 10 products, 10 RDCs, 5 CDCs and 10 source locations. It is analogous in size to the one which will be studied in more detail in section 9.1. The number of breakpoints was arbitrarily set to be equal to 5. The optimal objective value of the problem was 1971.

The algorithm was implemented using the Google OR-Tools Java wrapper ([Lau14]) for the mixed-integer programming solver CBC developed by the organization COIN-OR ([LH03]). It was run on an Intel Core 2 Duo CPU P8600 computer with 2.40GHz and 4GB RAM using a 64-bit Windows 7 operating system. No solution was found in the allotted time of ≈ 2.75 hours[5]. The lower bound found after this time was 1030, which is still 47% below the actual value of 1971. Since actual problems include transportation costs and thus multiple locations might be open in an optimal solution, it does not seem likely that linear optimization techniques can be used to solve this problem. This is one of the reasons motivating the use of a heuristic

6.2 Motivation for heuristic approach

Recently there have been several papers highlighting the effectiveness of heuristic evolutionary algorithms for facility location problems, such as [SOU09], [Tiw+10] and [CG+12].

[LG05] state that

> the common factor in meta-heuristic algorithms is that they combine rules and randomness to imitate natural phenomena. These phenomena include the biological evolutionary process, animal behavior and the physical annealing process.

The drawback of heuristics is that they are not guaranteed to find the optimal solution. In fact they may not even find a good solution or converge at all. However, to understand a real life distribution supply chain a ”sufficiently good” solution is frequently all that is needed, because both, the simplifications made during the modeling and the imprecise nature of the input data introduce significant unavoidable errors.

Real life supply chain networks are very complex. For this reason, assumptions must be made to create a sufficiently small model which can be solved within reasonable

[5]Limit was set to 10,000 seconds

time. But, because of the omitted details, an exact solution of the model is not an exact solution to the real world network. Examples of such simplifications are modeling the actual demands with an expected demand μ, and replacing multiple customers in geographic proximity to each other with one larger customer.

Moreover, the quality of input data frequently comes with an error margin of $\pm 10\%$ or more. Estimations are required to obtain future demands, transportation costs, potential growth rates and customer behavior in a changed network. Exact determination of the input variables is often not possible, because historic data is not guaranteed to be a good indicator of future events and predictions are, as the saying goes, "difficult, especially about the future".

Due to these two factors, an exact solution for a real world network which correctly predicts the future cost of a network, cannot be obtained from a model. However, the results of a good model should strongly correlate with the real network and be a good approximation of the costs. But when comparing the best and the second best alternative according to the model, the difference in the model costs might be smaller than the impact some of the omitted details may have. In such a case the decision could be tipped one way or the other.

For example, assume that the best solution is slightly better than the current network, but structurally completely different. The potential risks of completely overhauling a network, training new employees and the resistance within the organization would lead almost any manager to opt in favor of the current setup. Therefore, it would be highly advantageous, if the model could find not just the optimal solution to the model, but also a list of further "very good" ones. These very good solutions could then be evaluated "manually" based on those characteristics, which had been omitted in the model.

Since an exact solution is not required, heuristics become an attractive alternative. Moreover, it is easy to include more elements into the cost calculation, as one is not constrained to a linear formulation of the problem. Furthermore, in a evolutionary algorithm many feasible solutions are generated. It is then possible to explore the vicinity of promising solutions and find not just the best, but rather the desired list of "very good" solutions.

6.3 Heuristic approach to multi-echelon facility location problems

Before an approximative approach will be outlined in this section, let us look again at the three questions which the model attempts to answer.

a) Which distribution centers should be opened?

b) From which locations should customers be supplied?

c) How should flows between own locations be structured?

In the papers which do use heuristics to solve facility location problems, the model either represented a single-echelon network, e.g. [SOU09] and [CG+12], or a rather small network, e.g. [Tiw+10]. It is not necessary to simultaneously answer all three questions at the same time in a single-echelon network, because RDCs must receive all products via shipments from the source locations. Thus, the authors only had to look at the first two questions a) and b).

6.3.1 Justification for not considering the entire problem at once

Let us, for a moment, assume that variance and demand at the RDC locations were known. In this case only the third question c) remains to be considered, i.e. how to structure flows between sources, central distribution centers and regional distribution centers. In chapter 8 it will be shown that in the case of only one open central distribution center[6] and a small[7] number of RDCs this problem can be solved quickly by simple enumeration. However, the problem grows exponentially in the number of open RDCs, and therefore the enumeration approach is not scalable. To deal with cases in which a larger number of regional distribution centers is open, an efficient heuristic will be presented.

Knowing that the flow problem can be solved easily does not directly imply, how to answer all three questions simultaneously. But, as stated above, evolutionary algorithms have proven to be good at solving the single-echelon problem, i.e. answering questions a) and b). Once any set of open distribution centers is given as an answer to a) and any assignment of the customers to the distribution centers as an answer to b), then it is possible to quickly answer the problem of how products should flow from the source locations to the distribution centers.

Since good approaches exist to solving questions a) and b) and question c) independently and because the questions can be answered well in sequence, the proposed algorithm will not attempt to answer all three questions simultaneously. Improvements for the opening decisions and the customer assignment are found using a genetic algorithm, taking into account the total network cost. The cost of a setup is calculated as soon as customer assignment is known and the flow problem has been solved for this setup. The details of the genetic algorithm, such as the encoding and the evolutionary procedures, will be discussed in chapter 7 and the flow problem will be studied in more detail in chapter 8. In the following section the focus lies on explaining how the elements of the algorithm interact.

[6]If there are no central distribution centers, then the problem is trivial, because each regional distribution center sources each product directly from the source location

[7]Small in this context means less than ten

6.3.2 Algorithm

A formalized version of the algorithm is given below. In the following, an overview of each step is given along with a reference to the section, in which it will be discussed in more detail.

```
Create random network setups
while Termination criteria not met do
    for all Network setups do
        1. Assign customer demands to open locations based on assignment value
        2. Optimize flow from sources to RDCs, either directly or through CDC
        3. Calculate total cost of the setup
    end for
    Select low cost setups and perform crossover
    Reintroduce elite setups into the pool of potential setups
    for all Network setups do
        Apply small random mutations
    end for
end while
```

The genetic algorithm attempts to create increasingly better sets of open distribution centers and better assignments of the customer demands. To do this, n initial setups are generated at random, specifying which distribution centers should be open and from which distribution center the customers are supplied. All subsequent network setups are derived from these initial solutions during an iterative process.

In each iteration all the setups at first undergo a three step analysis process. In an initial step the customer demands are assigned to the open distribution centers based on the assignment decisions of the answer. This will be explained in section 7.2.1. Then the optimal product flows through the network are determined using the process described in chapter 8. At this point, all decisions have been made and the cost of the setup can be computed, taking into account all the cost elements included in the objective function defined in 5.2.

Pairs of network setups are then randomly selected, with low cost setups being more likely to be chosen. The opening decisions of distribution centers and the assignments of the two pairs are then recombined to create two new network setups. In this process, called *crossover*, a new group of $m \leq n$ setups are created. It will be discussed in more detail in section 7.2.2. Additionally, a group of $n - m$ previously found *elite* setups is reintroduced into the process, as described in section 7.2.3.

Then, each of the new setups is subjected to a few random changes, so called *mutations* to maintain a diverse set of setups and avoid early convergence. The rules governing mutation will be laid out in section 7.2.4.

All the steps relating to the genetic algorithm will be discussed in the next chapter and the flow problem in chapter 8. Cost calculation is done according to the objective function previously laid out in 5.2.

Chapter 7

Discussion of evolutionary aspects of the algorithm

Recently, several papers have been published, which used heuristics to obtain solutions for facility location problems with non-linear inventory costs. Examples are the papers by [CG+12], [DAO09], [LGR09], [MJKSK06], [SOU09] and [Tiw+10]. A frequently used type of heuristics are genetic evolutionary algorithms. The central idea behind genetic algorithms is based on Darwin's theory of evolution. In Darwin's theory a population adapts to its environment over the course of several generations due to a three step process.

1. **Selection of the fittest**
   The individuals within the population which are better adapted to their environment are more likely to survive and to propagate their genes to the next generation. In this way better genetic information is more likely to be passed on to future generations.

2. **Crossover**
   When two individuals produce offspring, the genetic information of the offspring is a combination of their parents' genetic information. Each child receives certain blocks of genetic information from one parent and other blocks from the other parent. In this manner an individual with distinct genetic information is created.

3. **Mutation**
   The genetic information of a child is not always just a recombination of its parents' genetic code. With a low probability some genetic information of the child will take a new value, irrespective of the value held by its parents. This change is called a mutation. Mutations are random and no thought goes into the selection, which parts change and to which value they change. The advantage of using random mutations is that potential improvements are generated, which had not been present in previous individuals.

Additional mechanisms such as "dominance, inversion, intrachromosomal duplication, deletion, translocation and segregation" were described by [Gol89]. The reintroduction of especially fit individuals of previous generations, called *elitism*, is a further mechanism which will be used in this thesis.

In the following section 7.1, the ideas and cocnepts underlying genetic algorithms will be presented. Then, these general concepts will be applied to the specific case of facility location problems in 7.2.

7.1 Theoretical background

The main concepts of evolutionary processes have been mentioned above. These concepts are included in genetic algorithms. A summary, what genetic algorithms are was given by [Gol89].

> Genetic algorithms are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search. in every generation, a new set of artificial creatures strings) is created using bits and pieces of the fittest of the old; an occasional new part is tried for good measure. While randomized, genetic algorithms are no simple random walk. They efficiently exploit historical information to speculate on new search points with expected improved performance.

The string structures, mentioned in this quote are the individuals in each generation. The defining information of each individual is contained within this string as a genetic code.

7.1.1 Encoding of possible solutions

The goal of optimization is to find a value $\hat{x} \in X$ which minimizes or maximizes the function $f(x),\ X \to \mathbb{R}$. Genetic algorithms work on encoding strings s, rather than on actual values $x \in X$.

Therefore, potential solutions x are broken down into their components and bijectively assigned to a unique string $s \in S$ by the bijection $g(x),\ X \to S$. The string s is the encoding of the value $x \in X$ and is defined over an alphabet A. Thus, for a string s of length l holds $s \in A^l$. A alphabet commonly used in genetic algorithms is the binary $\{0, 1\}$ alphabet, but a finite set of integers $\{1, 2, ..., a\}$ or the interval $[0, 1]$ are also used frequently.

A string s of length l is comprised of l individual elements $s_i \in A$. This means, that one can write $s = [s_1, s_2, ..., s_l]$. Each spot s_i in the string s is called a *gene* or *feature* and the value at that location in the string is the *allele* or *feature value*.

Since genetic algorithms only use the encodings $s \in S$ instead of the original variables $x \in X$, a new function $\bar{f}(s),\ S \rightarrow \mathbb{R}$ must be defined. For this function the property $\bar{f}(s) = f(g(x))$ must hold. The value $\bar{f}(s)$ is called the fitness value of s and by definition $x = g^{-1}(s)$.

Example 7.1
[Gol89] demonstrate this concept using the following problem as an example.

$$\begin{aligned} \max\ & f(x) = x^2 \\ \text{s.t.}\ & x \in X = \{0, 1, 2, 3, ..., 31\} \end{aligned}$$

A possible encoding of the set X over the alphabet $A = \{0, 1\}$ is to represent each number as a binary number. A string $s = [s_4, s_3, s_2, s_1, s_0]$ with $s_i \in A$ is mapped bijectively to a variable x according to

$$x = \sum_{i=0}^{4} s_i 2^i$$

For example, the string $\hat{s} = [0, 1, 0, 0, 1]$ represents the number

$$\hat{x} = 0 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 8 + 1 = 9$$

In the following table four initial random encodings, the number they represent and their fitness values are provided.

String s	Variable x	Fitness value $f(x)$
$[0, 1, 1, 0, 1]$	13	169
$[1, 1, 0, 0, 0]$	24	576
$[0, 1, 0, 0, 0]$	8	64
$[1, 0, 0, 1, 1]$	19	361

One can observe, that having $s_4 = 1$ tends to lead to a higher fitness function value than $s_4 = 0$. This is not surprising, as $s_4 = 1$ adds $2^4 = 16$ to the value of x and $f(x)$ is monotonously increasing. Since individuals with $s_4 = 1$ tend to be significantly better than those with $s_4 = 0$, these individuals are fitter and thus, more likely to propagate their genetic information to the next generation. Therefore, the information $s_4 = 1$ is likely to be more common in the next generation than it is in the current one. The goal of genetic algorithms is to identify key characteristics of "good" solutions and build strings which display many positive characteristics and few negative ones.

To describe these characteristics the concept of schema was introduced by [Hol93]. A schema is a string over the alphabet $A \cup \{*\}$. The $*$ character is a placeholder for any other element of the alphabet A. Assuming, again, a binary alphabet and a string of length 4, then the schema $[1, *, *, 0]$ would identify the set of all strings which have a 1 in the first spot, a 0 in the fourth and either a 1 or a 0 at the second and third spots. So, the four strings $[1, 1, 0, 0]$, $[1, 0, 0, 0]$, $[1, 0, 1, 0]$ and $[1, 1, 1, 0]$ all would be described by the schema $[1, *, *, 0]$.

Then [Hol93] goes on to state that highly fit schema with short defining length appear more frequently in later generations. These highly fit schema of short defining length are often called ”building blocks” and the statement that such schema become more prominent in later generations is called the ”building block hypothesis”, according to [Gol89]. The reason, why such schema appear more frequently lies in the fact, that genetic algorithms do not draw individuals with equal probability to propagate to the next generation. Rather, those with high fitness value are more likely to be selected for crossover. And these high fitness individuals tend to have more good building blocks at high leverage positions. Moreover, during crossover, they are unlikely to be broken up, because they are short.

7.1.2 Recombination of encodings

The two elementary ways used in almost all genetic algorithms to generate new potential solutions are crossover and mutation. An additional method used in this thesis, which can improve results, is elitism.

Crossover simulates the recombination of the parents' genetic information. Both parents' genetic information is encoded in a string of length l. During crossover a set of breakpoints $B = \{b_1, b_2, ..., b_k\},\ b_i \in \mathbb{N}$ is randomly generated with $0 < b_1 < b_2 < ... < b_k < l$. Between a breakpoint b_{i-1} and b_i the genetic information of parent p_1 is used and between breakpoints b_i and b_{i+1} the genetic information of parent p_2 is used. This is depicted in figure 7.1, where two parents produce two offspring individuals. In the parts of the encoding, where child c_1 has received the genetic information of parent p_1, child c_2 has received the genetic information of parent p_2. In those parts of the encoding, where c_1 has received the genetic information of p_2, the information of p_1 has gone to child c_2.

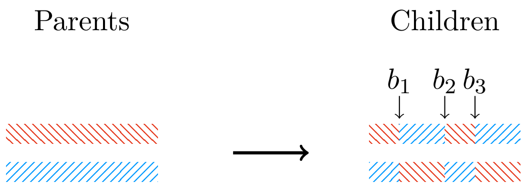


Figure 7.1: Genetic information of parents is recombined to create two children

Mutation, in contrast to crossover, is a purely random process. It does not maintain individual schema, but rather tries to break them and create new ones. Mutation impacts the allele, i.e. an individual value encoded in the string. A gene changes its feature value with a probability $p_{mutation}$. In a simple binary alphabet, each gene with value 1 would receive a value of 0 with probability $p_{mutation}$ or vice versa. In a more complex alphabet an additional probability must be defined, which determines which new value is assigned to the gene, if it is selected for mutation. Since mutations are purely random, many mutations do not improve results. However, occasionally a highly fit new schema is introduced to the gene pool. Moreover, mutation helps to maintain a diverse set of individuals.

An example is provided in figure 7.2. A binary string s of length 7 is given. Randomly, the fifth spot, shaded in blue, is chosen for mutation. The feature value at the fifth location then changes from $s_5 = 1$ to $s_5 = 0$.

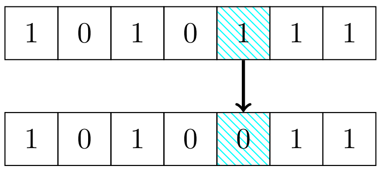


Figure 7.2: Mutation changes the value at a gene with probability p

Elitism is a way to steer the algorithm. The idea is to let a few high fitness individuals found in previous generations reenter the gene pool. Elitism ensures that the vicinity of good solutions is explored in more detail, because these individuals are frequently subjected to mutation and thus slightly different individuals are evaluated. If a current good setup is not yet a local optimum, then an even better setup lies in its vicinity. Since the inserted elite individual had a good fitness value, the mutated individual is also likely to be quite fit. Therefore, it has a high probability of propagating its genetic information. Thus, the building blocks of the elite individual are likely to be given to the next generation.

7.2 Proposed approach

In this section the encoding of the individuals and the crossover, mutation and elitism methods of the proposed algorithm are discussed. The set of all individuals in a generation k will be called $\mathbb{I}_k$. The set $\hat{\mathbb{I}}_k$ will denote the set of all individuals with a fitness value $f(i) < H$. The fitness value of all individuals which do not satisfy a constraint will be set to $H = 10^{15}$.

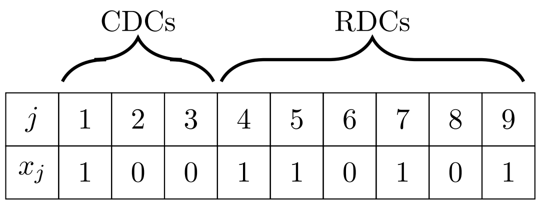
7.2.1 Encoding techniques

The genetic algorithm aims to tackle the two problems of which distribution centers to open and to which locations the customers are assigned. Therefore, two separate strings are used to encode the solutions to these problems. A binary string defines which distribution centers are open. An additional integer string specifies, from where customers are supplied.

Encoding of distribution centers

The binary string s^{dc} represents the set W of distribution centers. If the j-th entry $s_j^{dc} = 1$, then location $j \in W$ is open. If $s_j^{dc} = 0$, then location j is closed. Given the assumptions outlined in chapter 4, only one of the CDC locations $j \in W_{CDC}$ may be open. Also, limits on the number of open RDCs could be placed. Any solution which does not satisfy those constraints is automatically assigned an objective value H.

For an example of the encoding of distribution centers, refer to figure 7.3. The first three values represent CDC locations. Only the first value is equal to 1, while the second and third entries are equal to 0. Therefore, CDC 1 is open, while the other two CDCs are closed. The fourth through the ninth entries represent RDC locations. Since the depicted string has $s_4 = s_5 = s_7 = s_9 = 1$, the RDCs 4, 5, 7 and 9 are open, while the RDCs 6 and 8 are not open.



j	1	2	3	4	5	6	7	8	9
x_j	1	0	0	1	1	0	1	0	1

Figure 7.3: Example of distribution center string

Encoding of customer assignments

Furthermore, there is an integer string s, whose entries determine, to which distribution center a (product - customer) demand pair is assigned. Let d_n refer to the n-th (product - customer) demand pair. Then, an entry $s_n = j,\ j \in \mathbb{N}$ in the n-th position of the integer string s means, that the demand n is assigned to the j-th *closest* open system location. An upper limit m for the feature value s_n is defined, which specifies, how many of the closest locations could be considered.

In figure 7.4 an example of such a sample string is provided. The unique number in the first line refers to the (product - customer) pair. In the second line the assignment value is given. Lines three and four detail which product is demanded by which customer. These lines are not relevant for the encoding. According to the logic outlined above, the (product - customer) pair 1, representing the demand of product 1 at customer location 1 is supplied from the open system location which is closest to customer location 1.

Product - Customer	d_1	d_2	d_3	d_4	d_5	d_6	d_7	d_8	d_9
Assignment	1	2	1	1	1	1	1	1	2
Product	1	1	1	2	2	2	3	3	3
Customer	1	2	3	1	2	3	1	2	3

Figure 7.4: Example of (product - location) pair assignment string

In figure 7.5 the example given in figure 7.4 is laid out in a more graphical way. In the network three RDCs are open. The three distinct setups, divided by vertical lines, represent the situation for the three products. It can easily be seen, that customer location 1 is closest to location rdc_1 with rdc_2 being the second closest and rdc_3 being the farthest away.

Let us now consider in detail the situation for product 1. The (product - customer) pair d_1 has an assignment value of 1 and is consequently assigned to the closest location, which is RDC 1. In contrast, (product - customer) pair d_2 has an assignment value of 2. Therefore, the demand is not assigned to the RDC closest to customer 2, which would be RDC 2. Rather, it is assigned to RDC 1, the second closest RDC. The (product - customer) pair d_3 again has assignment value 1. Thus, it is supplied from the closest open location, which, in this case, is RDC 3.

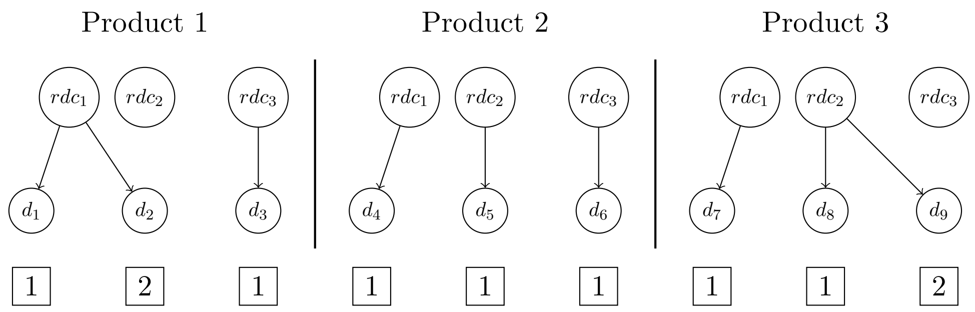


Figure 7.5: (Product - location) pairs are assigned to their n-th closest location. The parameter n is denoted in the box below each node.

It is important to reiterate, that two pairs with the same assignment value j are not necessarily supplied from the same location. Rather, they are assigned to the j-th closest location.

In some networks it is desirable, that certain products should always be supplied from the CDC. This can be incorporated into the encoding by using a dedicated assignment value, which always results in the (product - customer) pair being assigned to the CDC. Such a case is illustrated in figure 7.6. Assume that the limit is set to $m = 2$ and the CDC should always be considered. The two closest locations to the customer are RDC 1 and RDC 2. The feature, determining the assignment of this customer, could take on any feature value $s_i \in \{1, 2, 3\}$. If $s_i = 1$, then the customer would be supplied from RDC 1, since it is the closest. If $s_i = 2$, then it would be supplied from the second closest location, which is RDC 2. However, if the assignment value $s_i = 3$, then it would not be assigned to RDC 3, which is the third closest location. Rather, it would be assigned to the CDC, even though it is the location farthest away from the customer.

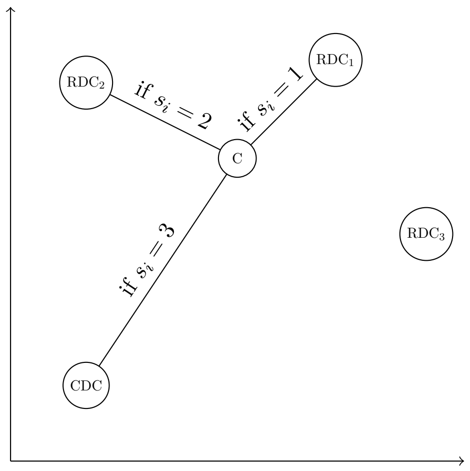


Figure 7.6: Assignment when considering the 2 closest locations and the CDC

7.2.2 Selection of fittest individuals and crossover

The selection process determines which individuals should advance their genetic information. The crossover process then determines, in which way they pass on that information to the individuals of the next generation. Therefore selection of the fittest and crossover are tightly linked.

The selection process is based on roulette wheel selection. Each individual is assigned a probability corresponding to the individual's objective value. Based on the probabilities, each individual i is assigned an interval $[a_i, a_{i+1}]$ with $a_i < a_{i+1}$ and $0 \leq a_i \leq 1$. If a random number r drawn from the interval $[0, 1]$ lies within the interval

$[a_i, a_{i+1}]$, then individual i is selected for crossover. This is analogous to a roulette wheel, where a number is the winning number, if the ball lands within the area, which is assigned to the number. The selection process is concerned with assigning each individual an interval of adequate size, based on the individuals fitness value.

In each generation $k \in \mathbb{N}$ the fitness value of each individual $i \in \mathbb{I}_k$ is computed. The fitness values of all individuals $i \in \hat{\mathbb{I}}_k$ with fitness values smaller than H, are added up. This results in a generational fitness value $T = \sum_{i \in \hat{\mathbb{I}}_k} f(i)$. Two methods to compute an individual's probability to be selected for crossover have been implemented. The first method assigns the probabilities only based on the individuals' fitness values. The second method assigns each individual a minimum probability and increases this probability based on the individuals fitness value.

For ease of notation two variables are defined.

$$m_k = \min\{f(i), i \in \hat{\mathbb{I}}_k\}$$
$$M_k = \max\{f(i), i \in \hat{\mathbb{I}}_k\}$$

The purely fitness based method selects an individual $i \in \mathbb{I}_k$ with probability

$$p_i^f = \begin{cases} \gamma(1 - \frac{f(i) - m_k}{M_k - m_k}) & \text{if } i \in \hat{\mathbb{I}}_k \\ 0 & \text{if } i \notin \hat{\mathbb{I}}_k \end{cases} \tag{7.1}$$

According to the density function p_i^f, the fitter individuals with lower cost have a higher selection probability. The scaling parameter $\gamma = (|\hat{\mathbb{I}}_k| - \frac{T}{M-m})^{-1}$ ensures that the probabilities sum up to 1. For all individuals $i \in \mathbb{I}_k \setminus \hat{\mathbb{I}}_k$ with an "infinite" fitness value $f(i) \geq H = 10^{15}$ the probability is set to $p_i^f = 0$.

The minimum probability method guarantees each individual a minimum selection probability $p_{min} = \frac{1}{|\mathbb{I}_k|}$. The actual probability p_i^m is then given as the sum of p_{min} and p_i^f, divided by 2. The scaling factor $\frac{1}{2}$ ensures that the sum of all probabilities is equal to 1.

$$p_i^m = \frac{1}{2}\left(p_{min} + p_i^f\right) = \frac{1}{2}\left(\frac{1}{|\mathbb{I}_k|} + p_i^f\right) \tag{7.2}$$

Using either the fitness based probabilities p_i^f or the minimum probabilities p_i^m, two parent individuals $i_{p_1}, i_{p_2} \in \mathbb{I}_k$ are selected for crossover. The two parent individuals produce two children $i_{c_1}, i_{p_2} \in \mathbb{I}_{k+1}$. Two sets of random breakpoints are generated for the distribution center string and the assignment string. Crossover of the parents' genetic information at these breakpoints creates the children's genetic strings.

7.2.3 Reinsertion of elite individuals

Elitism, as discussed previously, describes the idea of keeping or reintroducing highly fit individuals into the genetic pool. In the implementation a set containing the top ten[8] individuals with distinct distribution center strings is maintained. Thus, no two individuals in the top ten mandate exactly the same distribution centers to be open and closed. The restriction on individuals with distinct distribution center strings maintains diversity and avoids premature fixation of the algorithm on early solutions.

During an initial phase of 500 generations[9] elitism is not applied. In this phase the focus lies on natural exploration and finding a set of good configurations. After the initial phase, a few randomly selected individuals from the top ten are reinserted into the gene pool in each generation. This reinsertion happens prior to the mutation procedure and consequently, the elite individuals are also subjected to mutation.

7.2.4 Mutation of individuals

The children, produced by the crossover procedure, and the reinserted elite individuals all undergo mutation. During mutation the feature values of randomly chosen features in the distribution center string and in the assignment string are changed.

Since the distribution center string is a binary one, it changes its value from 0 to 1 or vice versa. Different probabilities are employed depending on the type of location. If an element represents a CDC, then the feature value is changed with probability p^m_{cdc}. In contrast, if it represents an RDC, the feature value changes with probability p^m_{rdc}.

The mutation procedure is more complex for the assignment string, because for each element of the string, it must be decided, if it is changed, and if yes, to which value. This second step is necessary, because each element can take on more than just two values. Therefore, a probability p^m_{assign} is defined, denoting how likely it is that an element of the string is selected for mutation. As described in section 7.2.1, a maximum number m is defined, specifying how many of the closest locations should be considered. In addition, the CDC location could always be an option to source from. If it is, then a mutated feature of the assignment string could also take on the feature value $m + 1$. Let

$$\hat{m} = \begin{cases} m+1 & \text{if the CDC should always be considered} \\ m & \text{otherwise} \end{cases}$$

To determine the new value of a feature, selected for mutation, a number is randomly chosen from the set $\{1, ..., \hat{m}\}$.

Example 7.2
Consider an example in which the three closest locations and the CDC should be considered for the customer assignment. Then, $\hat{m} = 3 + 1 = 4$. By chance, the third entry s_3 of the string s was chosen for mutation. The original value of $s_3 = 1$. A

[8]The number of individuals x to be used for elitism can be configured, but for the remainder top ten will be used since it is clearer than writing top x.

[9]Value of 500 generations was selected after multiple tests and could be different for other problems.

new value is assigned to s_3. It is randomly picked from the set $\{1, 2, 3, 4\}$. So, for example, the new value after mutation could be $s_3 = 2$. This means that the (product - customer) pair d_3, represented by feature s_3, is now supplied from the system location, which is second closest to it.

Chapter 8

Solution of non-linear flow problems

Once the customer demands have been assigned, the flow through the network must be optimized. For each product and each RDC there are only two options, which are depicted in figure 8.1. The CDC either is supplied directly from the source along the solid edge or, if a CDC is opened, it can be supplied via this CDC. In the second case, the product flows along the dashed edges from the source location to the CDC and from there to the RDC. The costs are compared to decide, from where each RDC should receive each product. These costs include transportation costs, handling costs and inventory costs. The unit transportation and handling costs can be determined. Inventory costs, on the other hand, are more problematic, because of their non-linear nature. However, the inventory costs at an RDC can be computed prior to the optimization in the two possible cases of being supplied from the CDC or from the source. Consequently, only the inventory costs at the CDC remain problematic.

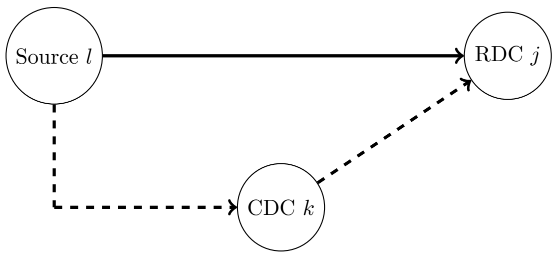


Figure 8.1: The RDC can be supplied directly from the source location or receive a product from the CDC

In section 8.1 the flow problem will be formulated. In section 8.2 three approaches to solving the problem will be presented. They will be compared in 8.3 and a mixed approach will be selected.

8.1 Problem description

In the previous chapter the assignment of demands to the distribution centers was discussed. These products must be shipped to the distribution center in order to satisfy the demands. These replenishment shipments can be made directly from the source location of the product, or it can be made from the central distribution center. If the replenishment is done from the central distribution center, then the CDC must have previously received the product from the source location itself and inventory must be kept at the CDC location. The cheaper of the two paths to supply a product to an RDC should be used.

Each product is produced in exactly one source location $l \in S$. It was assumed that there is at most one open CDC $k \in W_C$. In this chapter, it will be assumed that exactly one CDC is open. If no CDC were open, then the problem would disappear, as all the RDCs must be supplied from the source location in such a case. The set of open RDC locations is called $J \subset W_R$. The cost c_{lj} to supply an RDC j from a source l can be calculated using the inventory formulas discussed in chapter 3, and the linear transportation costs. The same can be done for the cost c_{kj} to supply the RDC from the CDC k. The cost c_{lk} cannot be determined a priori, because the quantity flowing through the CDC is not yet known. Based on the product flows the inventory cost $I(\sigma_k^2)$ is calculated analogously to the safety stock cost defined in (3.4).

$$I(\sigma_k^2) = (z^p \ \sqrt{\sigma_k^2}\sqrt{rlt_{lk}+1}) \cdot ihc^p$$

The linear transportation, handling and cycle stock costs per unit of product p along the edge from source l to CDC k can be calculated and are denoted by $\hat{c}_{lk}$. The setup can be seen in figure 8.2.

Since the cost along the entire path from source location l through CDC k to RDC j must be considered, the cost term $c_{lkj} = \hat{c}_{lk} + c_{kj}$ is defined. The actual cost of supplying the RDC via CDC k is larger than c_{lkj}, because safety stock costs $I(\sigma_k^2)$ are not yet included. For each RDC j the variable $q_j = c_{lkj} - c_{lj}$ is defined, representing the savings of the linear cost in the case of supplying RDC j from the CDC. Since c_{lkj} is a lower bound on the cost of supplying RDC j through the CDC, an RDC j will never be supplied via the CDC, if $q_j \geq 0$. So, the only locations which could potentially be supplied from the CDC k are those where $q_j < 0$. Let $\hat{J} := \{j \mid j \in J \text{ and } q_j < 0\}$ the set of all open RDCs which might be supplied from the CDC.

The problem now is reduced to decide which $j \in \hat{J}$ are supplied via CDC k. All terms in the objective function, except $I(\sigma_k^2)$, are linear. Thus, a mixed integer non-linear

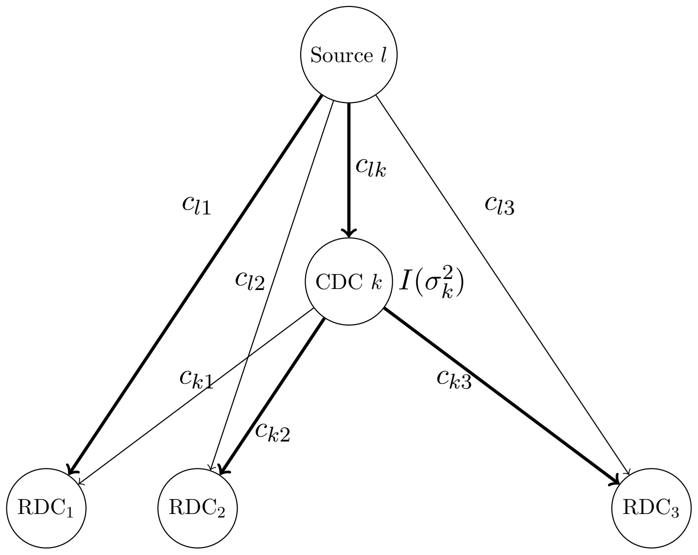


Figure 8.2: Flow network for one product with the cost terms

program can be formulated in the following way.

$$\min \ = \sum_{j \in \hat{J}} x_j q_j \ + I(\sigma_k^2) \tag{8.1a}$$

subject to

$$(\sigma_k)^2 = (\sigma_k^c)^2 + \sum_j x_j (\sigma_j)^2 \tag{8.1b}$$

$$x_j \in \{0, 1\} \qquad \forall j \in \hat{J} \tag{8.1c}$$

The objective function gives the cost saving of supplying the RDCs with $x_j = 1$ via the CDC, compared to supplying all directly from the factories. The constraint (8.1b) states that the variance at the CDC is the sum of the variances of the assigned RDCs and the sum of the variances $(\sigma_k^c)^2$ of the customers who are directly supplied from the CDC. The binary decision variables x_j indicate whether an RDC is supplied from the CDC or directly from the source.

$$x_j = \begin{cases} 1 & \text{if RDC } j \text{ is supplied from CDC } k \\ 0 & \text{if RDC } j \text{ is supplied directly from the source location } l \end{cases}$$

8.2 Description of the three approaches to the flow problem

To solve this problem, three approaches will be compared. The first and simplest solution procedure will be a simple enumeration algorithm testing all possible components. The enumeration approach has a runtime of $\mathcal{O}(2^n)$ with $n = |\hat{J}|$ being the number of open RDC locations with $q_j < 0$. For small problems, though, such a brute force approach is sufficiently quick, as will be seen in 8.3. Another approach to the problem is the use of a DROP heuristic. The third method presented will be a linear programming approach with a piecewise linear approximation of the concave square root term $\sqrt{(\sigma_k)^2}$ in the safety stock formulation $I((\sigma_k)^2)$.

8.2.1 Brute force enumeration of solutions

In the mixed integer non-linear program formulated above there are $n = |\hat{J}|$ variables. They can take on a value of either 0 or 1. This results in 2^n combinations of the n binary variables. If the cost of each combination has been calculated, then one can select the one with the lowest cost. This combination is the optimal solution. For each combination the function (8.1a) must be evaluated. Since there are 2^n combinations, the function must be evaluated 2^n times, which results in a runtime of $\mathcal{O}(2^n)$. For small n, the number of evaluations remains low, but it grows exponentially and soon the problem becomes intractable.

The approach has been implemented using recursive function calls to try all 2^n combinations.

```
function RecursiveFunction(array, currentLevel)
    if currentLevel = maxLevel then
        Evaluate(array)
    else
        array[currentLevel] ← 1
        RecursiveFunction(array, currentLevel+1)
        array[currentLevel] ← 0
        RecursiveFunction(array, currentLevel+1)
    end if
end function

function Evaluate(array)
    if Cost(array) < MinCost then
        MinCost ← Cost(array)
        optimalArray ← array
    end if
end function
```

A binary tree is created. In a node on level $j \in \{1, 2, 3, ..., n\}$ the function receives an array in which the first $j - 1$ entries have already been filled in. This function then calls itself twice for a node on level j, once with $x_j = 0$ and once with $x_j = 1$. This is repeated until $j = n$ and the bottom of the tree has been reached. At this point the function in (8.1a) is evaluated. If the found objective value is better than the best previously found one, the current solution replaces the previously found best one. Once all combinations have been evaluated, the combination which minimizes the problem is returned. The tree is modeled in figure 8.3.

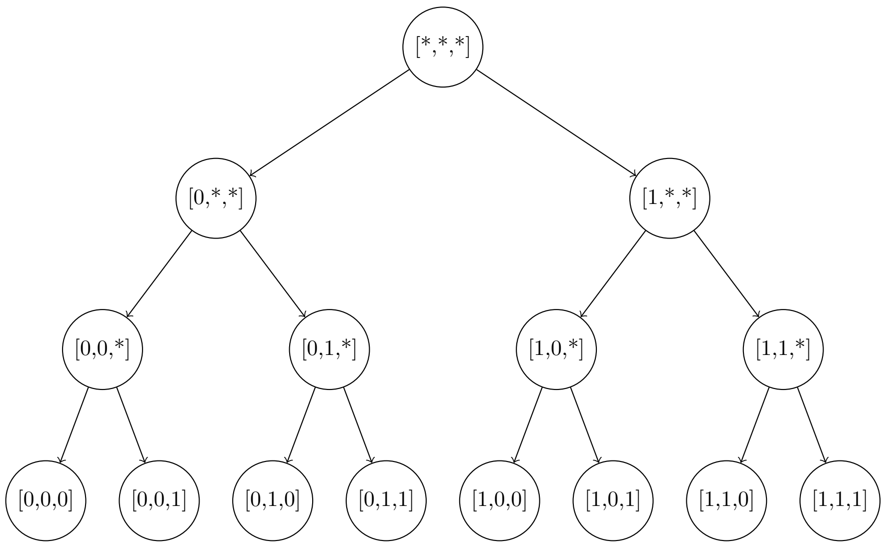


Figure 8.3: Tree created by the recursive brute force enumeration algorithm

It will be important to compare the runtime of this approach to the runtime required by the other algorithms and determine when it is advantageous to apply this brute force approach.

8.2.2 DROP heuristic

The DROP heuristic is a greedy algorithm. An initial set is given and elements are dropped from the set, if it directly improves the system. Since it is a greedy algorithm, it is quick, but does not always find the optimum. The application of such a heuristic to the flow problem will be explained in 8.1a.

In the initial step all RDC locations $j \in \hat{J}$, for which $q_j < 0$, have their corresponding variable $x_j = 1$. Given a combination of open and closed locations, for each open location j the costs of the two following cases will be compared

a) j is supplied from CDC k

b) j is supplied from supplier l

Whenever the cost in case b) is lower than in case a), then the location j is dropped from the set of locations, which are sourced from the CDC, and $x_j = 0$. This procedure is repeated until no location can be found such that dropping it leads to lower system-wide costs than continuing to source it from the CDC.

The process is best explained by looking at an example. A set of 3 RDCs is given with identical demand $\mu_j = \mu = 1$ and variance $\sigma_j^2 = \sigma^2 = 1$ at all three RDC locations $j \in \{1, 2, 3\}$. The linear cost savings q_j of supplying RDC j from CDC k, instead of directly from factory j, are given in the following table.

RDC	q_j
1	-7
2	-5
3	-1

The function $I(\sigma_k^2)$ can take the following three values.

$I(0)$	0
$I(1)$	5
$I(2)$	7
$I(3)$	8.5

Consequently, the cost of sourcing all locations through the CDC is calculated as

$$\sum_{j \in \{1,2,3\}} q_{kj} + I(3) = -7 - 5 - 1 + 8.5 = -4.5$$

The local improvement of dropping one location from being sourced from the CDC can be calculated as

Drop Location	Cost saving	Improvement
1	0	$+4.5$
2	-2	$+2.5$
3	-5	-0.5

Since dropping location 3 and sourcing it directly from the supplier reduces costs by 0.5 units, the location is better sourced from the supplier. The variable $x_3 = 0$ and the total cost saving is now -5 instead of -4.5. In the next iteration step the improvement of dropping one of the remaining locations is

Drop Location	Cost saving	Improvement
1	0	$+5$
2	-2	$+3$

Since both RDCs have a positive improvement value, they should continue to be supplied via the CDC. Thus, in the structure found by the DROP heuristic the RDC location $j = 3$ is supplied directly from the source location and the RDC locations $j = 1$ and $j = 2$ are supplied via the CDC.

An important advantage of the DROP heuristic is its quickness. The worst-case performance is $\mathcal{O}(n^2)$, but frequently it needs even less time. It also can be implemented easily. A large drawback is that it is a greedy heuristic and thus, does not always find an optimal solution.

8.2.3 Combinatorial optimization with piecewise linear function

Since non-linear optimization tends to be more difficult than linear optimization, a frequent technique to solve problems with a non-linear objective function and linear constraints is the use of piecewise linear functions. A non-linear function $f(x)$ can be approximated with linear segments $\hat{f}_i(x)$ on intervals $[p_{i-1}, p_i]$. The function $\hat{f}(x) = \hat{f}_i(x)$ for $x \in [p_{i-1}, p_i]$ is called a piecewise linear approximation of $f(x)$. Linear optimization techniques can be used to optimize a piecewise linear objective function, if additional binary decision variables and constraints are formulated.

The selection of the n points for interpolation impacts the precision of the piecewise linear approximation. In the flow problems a square root function $I(\sigma_k^2)$ must be approximated between $p_0 = 0$ and $p_n = \sum_j \sigma_j^2$. While there are sophisticated methods to select those points, choosing n equidistant points $p_i = i\frac{p_n}{n}$ with $i \in \{1, 2, 3, ..., n\}$ approximates the area under the curve reasonably well.

Lemma 8.1
Let $f(x) = \sqrt{x}$ on the interval $[0, a]$ and let $\hat{f}(x)$ be a piecewise linear approximation of $f(x)$. Moreover, let $p_i = i \cdot \frac{a}{n}$, $i \in \{1, 2, 3, ..., n\}$ and $p_0 = 0$ define the n intervals $[p_{i-1}, p_i]$ used to linearly approximate $f(x)$. Then, $f(p_i) = \hat{f}(p_i)$, $\forall i \in \{0, 1, 2, ..., n\}$. The area A_L under the piecewise linear function $\hat{f}(x)$ lies completely within the area A under the function $f(x)$. The percentage of the area A, which is not covered by A_L is given by the value δ with

$$\delta = 1 - \frac{A}{A_L} = 1 - \frac{3}{4}\frac{1}{n^{\frac{3}{2}}}\sum_{i=1}^{n}(\sqrt{i-1} + \sqrt{i}) \tag{8.2}$$

Proof. Determining the area A under the function $f(x) = \sqrt{x}$ on the interval $[0, a]$ can be done using integration. It is equal to $A = \frac{2}{3}x^{\frac{3}{2}}$.

The area under a line $g_i(x) = m_i x + t_i$ between two points $(p_{i-1}, f(p_{i-1}))$, $(p_i, f(p_i)) \in g_i$ is given by the formula $\frac{p_i - p_{i-1}}{2}(g_i(p_{i-1}) + g_i(p_i))$ according to the trapezoid rule. From the definition of the linear pieces g_i it is known that $g_i(p_i) = f(p_i)$ and $g_i(p_{i-1}) = f(p_i)$. Since the points p_i are equidistant, the term $p_i - p_{i-1} = d$ $\forall i$. Therefore, the area under the entire piecewise linear function

$$\begin{aligned}
A_L &= \sum_{i=1}^{n} \frac{p_i - p_{i-1}}{2}(g_i(p_i) - g_{i-1}(p_{i-1})\\
&= \frac{d}{2}\sum_{i=1}^{n}(f(p_i) + f(p_{i-1}))\\
&= \frac{d}{2}\sum_{i=1}^{n}(\sqrt{p_i} + \sqrt{p_{i-1}})\\
&= \frac{d}{2}\sum_{i=1}^{n}(\sqrt{i\frac{p_n}{n}} + \sqrt{(i-1)\frac{p_n}{n}})\\
&= \frac{p_n}{2n}\sum_{i=1}^{n}\sqrt{\frac{p_n}{n}}(\sqrt{i} + \sqrt{i-1})\\
&= \frac{p_n^{\frac{3}{2}}}{2n^{\frac{3}{2}}}\sum_{i=1}^{n}(\sqrt{i} + \sqrt{i-1})
\end{aligned}$$

Using this result, we get

$$\begin{aligned}
\frac{A_L}{A} &= (\frac{p_n^{\frac{3}{2}}}{2n^{\frac{3}{2}}}\sum_{i=1}^{n}(\sqrt{i} + \sqrt{i-1})) : (\frac{2}{3}p_n^{\frac{3}{2}})\\
&= \frac{3}{4}\,\frac{1}{n^{\frac{3}{2}}}\sum_{i=1}^{n}(\sqrt{i} + \sqrt{i-1})
\end{aligned}$$

From this result the equation follows directly. □

In the case of $n = 10$ the resulting $\delta < 0.01$ and thus the deviation is sufficiently small.

To reformulate problem 8.1a we introduce two additional sets of variables. Binary variables y_i and continuous variables y_i with $i \in \{1, 2, 3, ..., n\}$. We associate each y_i and z_i with the i-th segment of the piecewise linear function. In order to accurately describe the function the constraints must ensure that if $y_i = 1$ for any i, then $y_k = 1$, $\forall k < i$. Additionally, whenever $z_i > 0$ then $\sum_{k=1}^{m} z_k = b_m$, $\forall m < i$. This rule ensures that the first segments of the curve, which have a steeper slope than than the later ones, are maxed out, before a cheaper rate is used. Let $m_i = \frac{\sqrt{b_i} - \sqrt{b_{i-1}}}{b_i - b_{i-1}}$ be the slope of each segment. The linear problem then can be formulated in the following way.

$$\min \sum_{j \in \text{RDC}} x_j q_j \; + \sum_{i=1}^{n} z_i m_i$$

subject to the constraints

$$\begin{aligned}
(\sigma_k)^2 &= \sum_j x_j(\sigma_j)^2 \\
(b_i - b_{i-1})y_{i+1} \leq z_i &\leq (b_i - b_{i-1})y_i \quad \forall i = 1, 2, .., n-1 \\
0 \leq z_n &\leq My_n \\
\sum_{i=1}^{n} z_i &= (\sigma_k)^2 \\
y_i, x_j &\in \{0, 1\} \qquad \forall i, j = 1, 2, ..., n \\
z_i &\geq 0 \qquad \forall i = 1, 2, ..., n
\end{aligned}$$

The problem was implemented using the Google OR-Tools Java wrapper ([Lau14]) for the open-source mixed integer programming solver CBC developed by the organization COIN-OR ([LH03]). A variation has also been implemented using SOS2 constraints. However, since the Java wrapper does not allow to pass on such constraint, they must be implemented manually and did not produce a better result.

8.3 Comparison and conclusion

The approaches described in 8.2.1, 8.2.2 and 8.2.3 have different runtimes and differ in their solution quality. In order to determine the efficiency of the algorithms the runtimes and errors are compared for different numbers of open RDCs in 8.3.1. A dual method combining the enumeration approach for small problems and the DROP heuristic for larger ones will be found to work very good, as will be described in 8.3.2.

8.3.1 Comparison of approaches

In the following at first the runtimes are compared and then the deviation in the case of using non-optimal methods such as the DROP heuristic or piecewise linear approximation are contrasted.

For each problem size each algorithm solves a number of randomly generated problem instances. The average runtimes in milliseconds (ms) for one problem instance can be seen in the next table. The time is the average it took a large number of problems. The number of problems depended on the approach, but was chosen in such a way that the time required to solve all problems was at least several seconds. This circumvents the problem of computers to measure very short time intervals. A graphical representation of the runtimes is given in figure 8.4. The y-axis is on a log scale, to accurately depict the rapid increase in solution time in the case of the DROP heuristic.

Number RDCs	Enumeration	DROP Heuristic	Piecewise Linear
2	$2.3 \cdot 10^{-4}$	$1.2 \cdot 10^{-4}$	13.7
4	$9.1 \cdot 10^{-4}$	$2.5 \cdot 10^{-4}$	17.8
8	$1.3 \cdot 10^{-2}$	$5.6 \cdot 10^{-4}$	21.0
16	3.8	$9.5 \cdot 10^{-4}$	26.2
32	$2.5 \cdot 10^{5}$	$2.2 \cdot 10^{-3}$	37.4

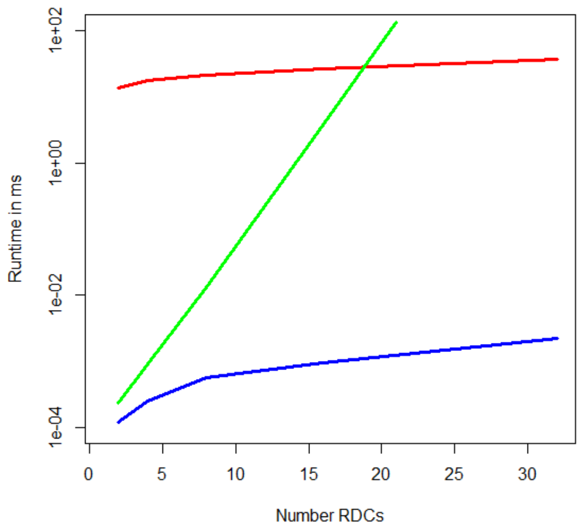


Figure 8.4: Y-axis represents runtime in [ms] on a log-scale while x-axis represents the number of RDCs. Steeply increasing green line represents brute force enumeration, low blue line represents DROP heuristic and high red line the piecewise linear algorithm.

As expected the brute force enumeration algorithm doubles its runtime with each additional RDC. For 32 RDCs the solution time is already in the range of 4 minutes. On the logarithmic scale in figure 8.4 the straight line indicates exponential growth of the runtime required in a brute fore enumeration. The runtime for the other two approaches grows significantly slower, but on very different levels. The red graph shows the average runtime for one run using a piecewise linear approach. A large part of this solution time can be attributed to setting up the connection to the solver and the initialization of the model. Even in the case of 32 RDC locations the DROP heuristic (blue line) can solve up to 500 subproblems per millisecond.

In the next table we see the difference on the savings. The percentage value does **not** represent the difference between actual costs, because only the cost benefits q_j are considered in the calculation[10]. The brute force enumeration algorithm is used as the benchmark since it always finds the optimal solution. Each algorithm solved the same $k = 10,000$ problems for the varying number of locations. For the two approximative approaches an approximation value f_i^{approx} was calculated and compared with the exact value f_i^{exact}. The total deviation $\delta = \frac{\sum_{i=1}^{k} f_i^{approx} - f_i^{exact}}{\sum_{i=1}^{k} f_i^{exact}}$ is computed across all iterations.

Number Locations	DROP Heuristic	Piecewise Linear
2	0.0%	−0.1%
3	8.0%	−0.3%
4	7.4%	−0.2%
6	3.5%	0.0%
8	1.9%	0.0%
12	1.2%	0.0%
16	1.0%	0.0%

As expected, the piecewise linear estimation is very accurate with a deviation lower than -0.4% and for medium number of locations lower than -0.05%. The reason for the decrease in the difference lies in the fact that the linear pieces increase the $\sqrt{x}$ function better, the larger x gets. Thus, as more variances are pooled together the solutions become more and more exact. The value $f_i^{approx} \leq f_i^{exact}$, because the square root function is concave and thus any line segment connecting two points on the function lies below the function. Therefore, all deviations are negative.

The deviation, when using the DROP heuristic, is less predictable. For a small set of 2 locations the heuristic yields very good results. However, in the case of 3 RDC locations, the deviation becomes relatively large. As the number of RDC locations grows, this deviation decreases again. A main reason for the poor performance, when the number of RDCs lies between three and six is the greedy nature of the heuristic. The concave nature of the safety stock function leads to a very small saving, when the first RDC should be dropped. The small safety stock saving often does not offset the larger saving q_j. However, if multiple locations had already been dropped, then the algorithm would often continue to drop even more, because then the safety stock savings are larger. As the number of locations gets larger than six, a different effect kicks in. Since the variances start adding up, it becomes more and more attractive to supply an RDC via a CDC, because the additional cost is almost negligible. Thus, when very few locations ought to be dropped, the DROP heuristics tendency to utilize the CDC becomes the right strategy.

The deviations are graphically depicted in figure 8.5.

[10]Savings must be considered in relation to total costs. Saving 1 Euro from total costs of 2 Euros is a significantly larger change than saving 1 Euro given total cost of 1000 Euros.

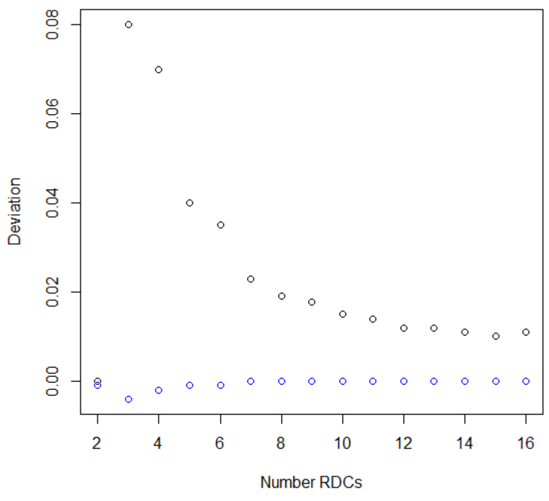


Figure 8.5: Drop heuristic has increased precision as number of locations grows. Piecewise linear solutions are always very close to optimal.

8.3.2 Selection of best method

Assume a set of 10 products, 4 open RDCs and 100 individuals per generation of the genetic algorithm. Then $N = 10 \cdot 100 = 10^3$ flow problems must be solved, with each one requiring $T_{it} = 17.8ms$ to be solved. Thus, solving all the N flow problems of one generation would take $T = N \cdot T_{it} = 10^3 \cdot 17.8ms = 17.8s$. This is too much time, since the algorithm needs to go through several thousand generations and even for a mere $1,000$ generations, almost 5 hours would have to be spent only on the flow problems.

Runtime can also become a problem with the brute force enumeration algorithm, as the number of open RDC locations increases to ten and beyond. However, for small problems in the range of 2 to 8 the required time is still relatively short and the approach can be used. Thus, for such a small number of RDCs it is reasonable to use the exact enumeration approach, especially because the DROP heuristic performs poorly in that range.

Once the problem becomes larger the DROP heuristic provides a better trade off between a small error and a still very short runtime. Therefore, in the experiments in the next chapter, problems with 8 or more RDCs will be solved using the DROP heuristic and smaller ones with the enumeration approach. For comparison, the results will also be compared to those of the purely DROP heuristic approach.

Chapter 9

Computational Results

Since the results obtained from a heuristic are very unlikely, and never with certainty, optimal, the quality of the solutions must be checked. To do this, multiple questions should be answered.

- **How close is the solution to the optimal solution?**
  The gap between the best found solution and the optimal solution should be as small as possible.

- **How stable is the solution?**
  Genetic algorithms always depend on an element of chance. However, a stable algorithm should generate similar solutions in each run and converge towards solutions with a similar objective value.

Two problem instances of different sizes will be compared in this chapter. One is a small manually generated problem (M) with 100 customer clusters, 10 potential RDC locations and 5 CDC locations. The 10 products are produced in 5 factories. This yields a total of 1000 (product - customer) pairs and 20000 (prouct - customer - sourcing location) triples. Multiple tests with regard to the parametrization will be performed on the test instance (M). These tests look at the effect of increases in inventory cost and how changes to the flow problem logic and crossover probability calculation impact the algorithm.

Additionally, the algorithm will be tested on a large real-life problem set (R) containing 387 customers, 20 RDCs, 6 CDCs and 239 different products coming from 18 factories.

9.1 Randomly generated problem instance (M)

Based on discussion with practitioners the assumptions for a simplified model were laid out. The model (M) was to include varying demands across the map and a varied product spectrum. It was randomly generated. The parameters will be outlined in the following.

Customers are located around the points $(\pm 4, \pm 4)$ by adding random numbers with distribution $N(0,1)$ to these four points. The RDCs were then distributed randomly with each coordinate being given by $X + sgn(X)$ and $X \sim N(0,2)$. The CDC locations were placed closer around the center with distribution $N(0,2)$. Factories were randomly distributed across the entire plane with distribution $N(0,3)$.

Such a network is depicted in figure 9.1. In the figure white circles represent customers. The RDCs are depicted as blue circles, the CDCs ad red diamonds and the source locations as green triangles. The locations are referenced by the number next to them.

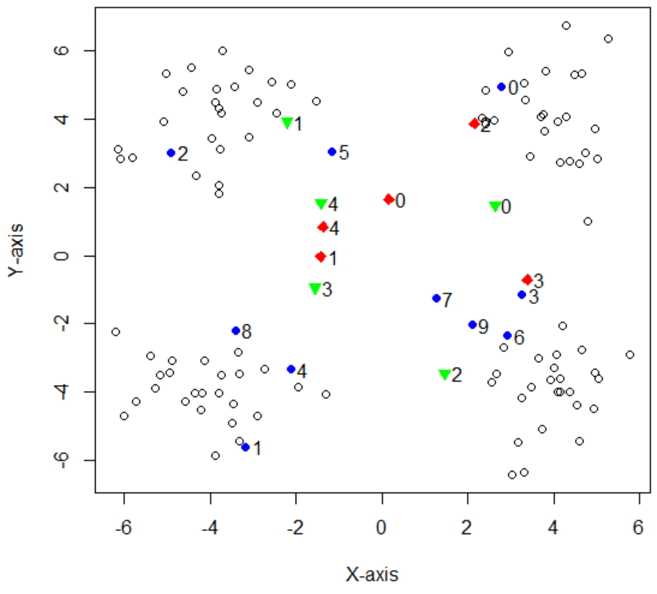


Figure 9.1: Example of a randomly generated network

The operating cost of a CDC location k is given by the function

$$fwc_k = f(x_k) = \frac{10^5}{(x_1^k)^2 + (x_2^k)^2}$$

and the operating cost of an RDC location j is given by the function

$$fwc_j = f(x_j) = \frac{2(10^4)}{(|x_1^j| - 4)^2 + (|x_2^j| - 4)^2}$$

Thus, a central warehouse is cheaper the further it is away from the center of the map. Analogously, a regional warehouse is cheaper, the further it is away from one of the four concentrations of customer locations.

In a next step, random demands were generated based on product and quadrant. This attempts to model regionally differing product demands and strength of economic activity. The weekly base demand μ_i^b of a product i depends on a random variable $X \sim N((i-1) \bmod 5 + 1, 10)$. The number of order lines per year is assumed to be given by a random variable $Y \sim Pois(5)$ and the coefficient of variation $\frac{\sigma}{\mu} = 3$ for all locations.

Products 1 to 5 have *high* demand at locations in quadrants one and two, while products 6 to 10 have *high* demand at locations in quadrants three and four. High demand in a region means, that the base demand rate is multiplied with a factor of 2. Analogously, quadrants one and three have strong economies, which means, that the demand in these regions is again multiplied with 2. Thus, product 1 would have an expected weekly base demand of $\mu_i^1 = 4$ at locations in quadrant one, while it would only have an expected weekly base demand of $\mu_i^4 = 1$ at locations in quadrant three.

In a next step, for each product i the characteristics cost, weight, source factory, lead time and demand class were defined. We let products i and $i+5$ for $i \in \{1,2,3,4,5\}$ be identical with regards to their properties.

Product	Cost	Weight	Source	Lead time	Demand Class
1	16	2	1	2	1
2	8	1	2	2	1
3	4	0.5	3	2	2
4	2	0.25	4	4	3
5	1	0.125	5	8	3
6	16	2	1	2	1
7	8	1	2	2	1
8	4	0.5	3	2	2
9	2	0.25	4	4	3
10	1	0.125	5	8	3

Distance d_{ab} between any two locations a and b is given by the euclidean distance metric. Given a shipment weight w the freight costs are calculated based on the formula $fc = d_{ab}\sqrt{w} + 1$. Cost of an outbound shipment is equal to 0.5 at all locations.

9.1.1 General behavior of algorithm

In the following the time required to find a solution, the precision and the stability of the algorithm with respect to problem set (M) will be studied.

The heuristic quickly finds good solutions. In figure 9.2 the average objective value in the case of inventory holding cost of 0.5 over ten runs of the algorithm is depicted[11].

[11] Only the value of every 10th generation is depicted for display reasons

It can be seen that in the first iterations large improvement steps are still being made, but later on there are fewer and smaller improvements. In the first few iterations a cluster of solutions forms around the value 108, 000. The subsequent improvements are a result of the introduction of elitism and thus, focusing in more detail on the already found ”good” solutions.

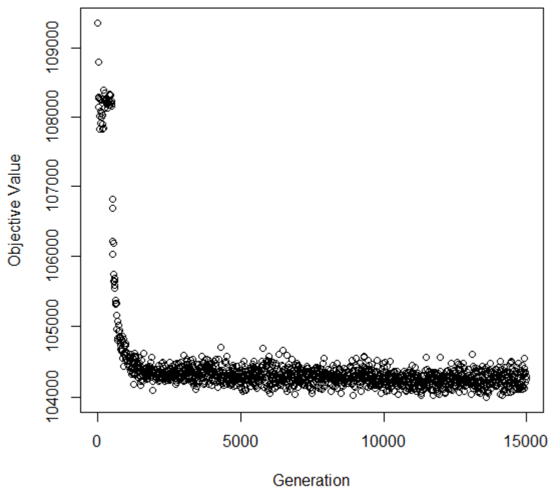


Figure 9.2: Graph of objective value of best individual in a generation.

Stability of solution

The algorithm for the same problem (M) was run ten time. The objective value across all ten runs was almost identical. The best individual across all ten iterations had a cost of 103, 715 Euros while the ”worst best” individual had a cost of 103, 736. This is a difference of just 0.02%. In addition to the found objective values not deviating much, it was also observed that the found solutions are very stable.

When looking at the ranking of the top ten one sees the same order of best to eighth best in every run, with the exception of two individuals with similar objective values swapping places in one run. This indicates that the algorithm is highly stable.

Required solution time

The algorithm was run on a Intel Core 2 Duo CPU P8600 computer with 2.40GHz and 4GB RAM using a 64-bit Windows 7 operating system. The maximum number of generations was fixed at 25, 000 and the algorithm was to terminate prior to this maximum if no improved individual had been found for 5, 000 generations. It took approximately 376 seconds or 6.26 minutes. Thus, the ten runs take slightly more than one hour. The average number of iterations was $22,707$ with 7 runs terminating due to the non-improvement for $5,000$ generations. The computations necessary for one generation took on average 16.54ms.

9.1.2 Effect of inventory holding cost

The effect of inventory holding cost impacts the structure of the solution. The algorithm was again run ten times for $25,000$ iterations each on the instance (M) with an inventory holding cost factor of 0.1, 0.5, 1 and 5. The results of the experiments are summarized in the following table.

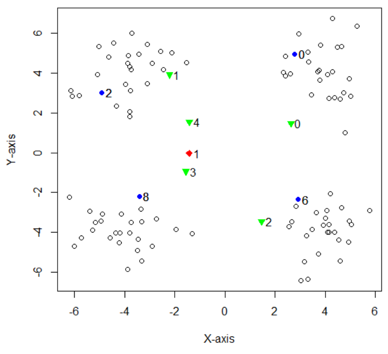
Inventory holding cost	Share of inventory cost	Average number of open RDCs
0.1	9.7%	5.12
0.5	28.3%	3.37
1	30.7%	2.15
5	63.0%	0.5

As expected, inventory costs increase as a share of total cost from 9.7% to 66.0% as the inventory holding cost factor increases from 0.1 to 5. At the same time the average number of open RDCs within the top ten individuals decreases from 5.12 to 0.5. Here it is important to notice that all top 5 solutions only had the CDC and no RDC. Since there are only 5 possible combinations with no RDC, the remaining 5 top ten locations had to have at least one.

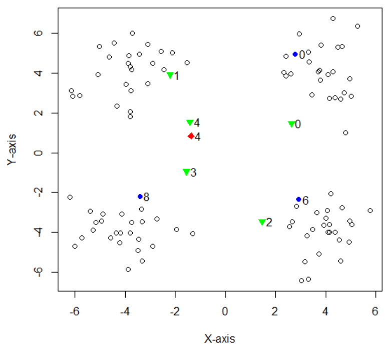
The fittest individual out of all twenty runs is depicted in figure 9.3. In figure 9.3a the best solution found was to open one CDC in the center of the map and one RDC in each of the four regions. As inventory holding costs increase, RDCs are being closed. In 9.3b there is no longer an RDC in the upper right quadrant. Rather, the CDC moves slightly higher and supplies that region directly. This trend continues in 9.3c. As the inventory holding cost factor grows to unrealistic heights, only the CDC at location 1 is used. In figure 9.3d it can be seen that no RDCs are opened.

9.1.3 Effect of changes in algorithm parameters

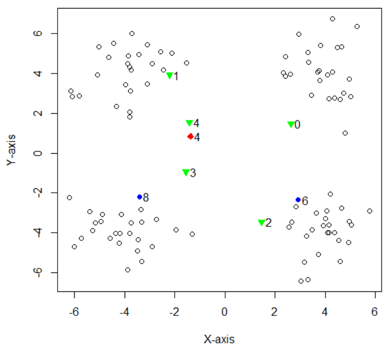
In the following, the impact of using only the DROP heuristic for solution of the flow problem is studied. Also, the convergence of individuals in the case of a purely fitness based crossover selection as opposed to one with a minimum probability, discussed in 7.2.2, will be compared.



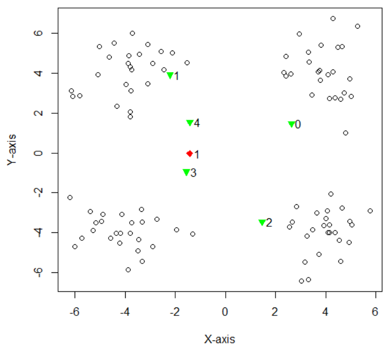
(a) Inventory holding cost factor of 0.1



(b) Inventory holding cost factor of 0.5



(c) Inventory holding cost factor of 1



(d) Inventory holding cost factor of 5

Figure 9.3: The number of open distribution centers decreases as inventory holding costs increase from figures 9.3a through 9.3d.

Comparison of DROP heuristic versus the combination of DROP heuristic and brute force enumeration

The impact of only using the DROP heuristic as opposed to using the exact brute force enumeration schema was negligible in the problemset (M). In the case of an inventory holding cost factor of 0.5 the objective value was identical in both cases, being set to be on average $103,721$. Both algorithms produced the same seven best individuals in the same order. Runtime also was not affected greatly. The time required for one iteration dropped from 16.54ms per generation when using the enumeration approach by only 1.4% to 16.30ms. However, the average number of iterations increased by 5.4%. The increase in the number of iterations is likely due to chance. A statistically significant amount of runs would have to be conducted. Since the impact is low either way, this problem was not studied in greater detail.

Comparison of purely fitness based crossover selection versus minimum fitness approach

As described in 7.2.2, two crossover probabilities were defined. The purely fitness based probability p_i^f is heavily skewed towards the better individuals and assigns high-cost individuals a very low probability. In contrast, the minimum crossover probability $p_i^m = \frac{1}{2}(p_i^f + \frac{1}{|I_k|})$ assigns each individual a probability of at least $\frac{1}{2|I_k|}$ with $|I_k|$ being the number of all individuals in a generation.

In figure 9.5 the difference between the average best value in each generation[12] across ten runs is depicted. It can be seen that the purely fitness based approach outperforms the minimum probability approach. In the initial stages prior to the introduction of elitism the difference is significant. Then the cost difference shrinks. However, it can be seen that the average best value using a minimum probability lies between 0 and 500 Euros higher than in the case of purely fitness based values. This corresponds to a 0% to 0.5% higher average cost.

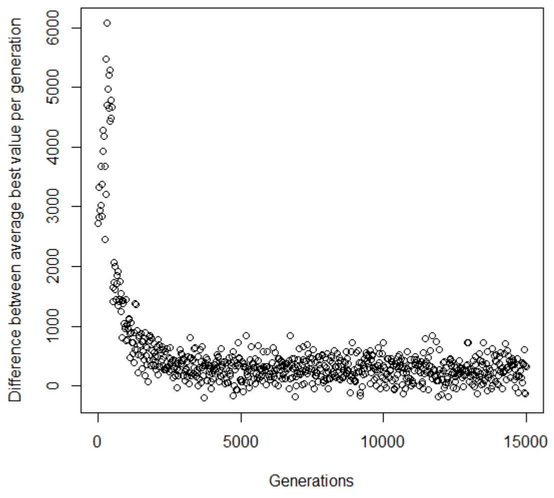


Figure 9.4: Average higher cost of best individual of minimum probability approach per generation

A similar, but less drastic, effect can be observed in figure 9.5 when looking at the overall best value found so far. The difference between the two values does converge

[12]Only value of every 20th generation is shown for display reasons.

to zero, but it takes the minimum probability approach longer to converge. This also results in fewer early terminations. With the fitness probability approach three of the ten runs did not terminate prior to the full 25,000 generations. This number goes up to eight of ten in the case of minimum probabilities.

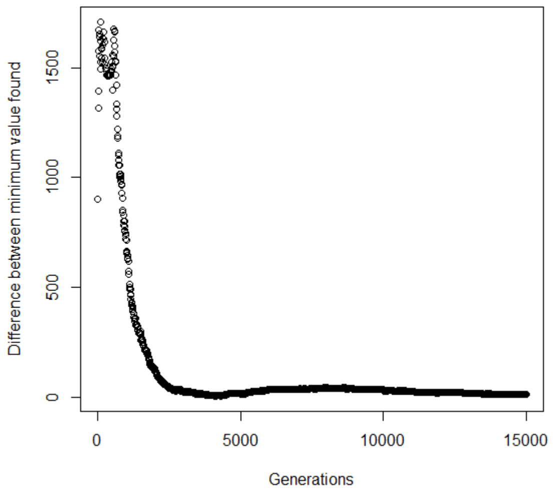


Figure 9.5: Average difference of best value found for minimum probability and fitness based probability per generation

9.2 Real-life problem instance (R)

In this section a real-life problem instance will be studied. This model includes 16,257 (product - customer) pairs which represent demand at 389 distinct customer locations for 239 distinct products[13]. There are 20 RDC locations and 6 potential CDC locations. There are 18 source locations at which the products are manufactured. Inventory holding cost is set at 0.34 and the α-service level is set to 98% for A parts, 95% for B-parts and 90% for C-parts.

With regards to the algorithm parameters, the maximum number of iterations is kept at 25,000 and the maximum number of iterations without improvement is set at 5,000. The number of individuals was decreased to 50 to speed up the algorithm. Each location gene is mutated with a probability of 0.05 and each assignment gene

[13]Most customers do not have demand for all products. Therefore, the number of (product - customer) pairs is a smaller number than the product of the number of products and the number of customer locations.

is mutated with a probability of 0.01. The crossover probability is purely based on individual fitness p_i^f. The algorithm was again run ten times.

The total cost was equal to 19.442 million Euros in all ten runs. Again, the solutions proved to be very stable, as the order of the top ten individuals was identical over all ten runs. The optimal solution for the problem had one CDC and an additional RDC. In the second best solution an additional RDC was opened.

It took on average $9,360$ generations to arrive at the solution. This corresponds to approximately 172 minutes. Thus, the calculations in one generation required 1.1 seconds.

Chapter 10

Conclusion and Outlook

In this thesis a mixed-integer non-linear program was formulated to extend the traditional facility location - allocation problem in multiple directions. Different flows of multiple products were modeled, which increased the problem size. The problem also became significantly more complex, because two levels of distribution centers were considered. The extension which changed the problem the most was the introduction of non-linear inventory costs. In combination with the extension to a multi-echelon network, the problem became too complex to be solved with linear optimization techniques.

To still obtain good solutions for this mixed-integer non-linear program a genetic algorithm heuristic was proposed. The opening decision of the facilities and the assignment of customers was determined for all the individuals of a generation using the genetic information. Then the optimal product flow between non-customer locations was calculated. This problem was simpler, because customer demands had already been assigned to the distribution centers. After the cost of all individuals had been calculated, three tools of genetic algorithms, crossover, mutation and elitism, were used to generate the new, and hopefully better, individuals of the following generation.

Three algorithms to solve the product flow problem were compared. The algorithms were a brute force enumeration approach, a DROP heuristic and a piecewise linear approximation. A combination of computationally inefficient, but exact, brute force enumeration for small problems and a greedy heuristic for larger problems was found to be quite effective.

The algorithm was then tested on a small, randomly generated problem instance (M). On this instance various tests were conducted to examine the influence of changes to modeling and algorithmic parameters. It was found that the algorithm was very stable and was able to obtain the solution within a very short time of just a few minutes.

Additionally, the algorithm was tested on a real-life problem (R) with a large number of distinct products and more than 300 customer locations.

The model could be extended in multiple directions. In realistic scenarios transportation costs are not necessarily linear in the quantity. Rather, they display economies of scale and are concave in the total quantity. Moreover, the freight mode, especially whether a shipment is send as a pallet or as a small parcel, greatly impacts transportation cost. Freight mode selection is very complex, but could make the model more realistic.

The restriction on only one CDC, while frequently sensible, does not always hold. Especially in global networks there may be two or three distinct central distribution centers located in the Americas, the Asia-Pacific area or in Europe. A larger number of CDCs would greatly complicate the flow problem. The DROP heuristic would have to be modified. Such an extension would have an even bigger impact on the enumeration approach, because an additional sourcing option would change the runtime from $\mathcal{O}(2^n)$ to $\mathcal{O}(3^n)$. Thus, a new approach to the flow problems would have to be developed.

Appendix

To run the genetic algorithm an installation of Eclipse, available at www.eclipse.org must be downloaded and installed. When starting eclipse the folder ”Genetic_Algorithm”, which is included in the attached DVD, must be selected as the workspace. In eclipse open the file ”Run” and set the string variable ”directory” to the path containing the desired input folder. This can be one of the folders

- ”Genetic_Algorithm_MA”
- ”Genetic_Algorithm_Large”

In order to run the piecewise linear parts of the algorithm it is necessary to install the Google OR-tools software [Lau14] available at http://code.google.com/p/or-tools/. Follow the instructions provided under the link ”Getting started” to install the Java wrapper and the CBC solver. To run the experiments detailed in the flow problems, open the workspace ”System_Location _Assignment _Algorithms”. Right click on the project and click on the menu ”Configure Build Path”. There, select ”Add External JARs” and select the file ”jnilinearsolver.dll” located in the folder \lib in the Google OR-tools folder. Then, the code can be run from Eclipse.

List of Figures

Bibliography

[ABX14] R. G. Askin, I. Baffo, and M. Xia. "Multi-commodity warehouse location and distribution planning with inventory consideration". In: *International Journal of Production Research* 52.7 (2014), pp. 1897–1910. ISSN: 0020-7543.

[Ama] Amazon.com. *Leipzig Logistikzentrum.*

[BAF12] A. Boloori Arabani and R. Z. Farahani. "Facility location dynamics: An overview of classifications and applications". In: *Computers & Industrial Engineering* 62.1 (2012), pp. 408–420. ISSN: 03608352.

[Baj88] C. Bajaj. "The algebraic degree of geometric optimization problems". In: *Discrete & Computational Geometry* 3.1 (1988), pp. 177–191. ISSN: 0179-5376.

[BDS98] A. I. Barros, R. Dekker, and V. Scholten. "A two-level network for recycling sand: A case study". In: *European Journal of Operational Research* 110.2 (1998), pp. 199–214. ISSN: 03772217.

[Bow13] D. J. Bowersox. *Supply chain logistics management.* 4th ed. New York: McGraw-Hill, 2013. ISBN: 978-0-07-132621-6.

[CBL96] J. J. Coyle, E. J. Bardi, and C. J. Langley. *The management of business logistics.* 6th ed. Minneapolis/St. Paul: West Pub. Co., 1996. ISBN: 0-314-06507-5.

[CG+12] G. Cabrera G., E. Cabrera, R. Soto, Rubio, L. Jose Miguel, B. Crawford, and F. Paredes. "A Hybrid Approach Using an Artificial Bee Algorithm with Mixed Integer Programming Applied to a Large-Scale Capacitated Facility Location Problem". In: *Mathematical Problems in Engineering* 2012.6 (2012), pp. 1–14. ISSN: 1024-123X.

[Coo63] L. Cooper. "Location-Allocation Problems". In: *Operations Research* 11.3 (1963), pp. 331–343. ISSN: 0030-364X.

[Coy09] J. J. Coyle. *Supply chain management: A logistics perspective.* 8e [ed.] Mason and OH: South-Western Cengage Learning, 2009. ISBN: 978-0324376920.

[DAO09] A. Diabat, T. Aouam, and L. Ozsen. “An evolutionary programming approach for solving the capacitated facility location problem with risk pooling”. In: *International Journal of Applied Decision Sciences* 2.4 (2009), p. 389. ISSN: 1755-8077.

[EM00] S. J. Erlebacher and R. D. Meller. “The interaction of location and inventory in designing distribution systems”. In: *IIE Transactions* 32.2 (2000), pp. 155–166. ISSN: 0740817X.

[Gol89] D. E. Goldberg. *Genetic algorithms in search, optimization, and machine learning*. Reading and Mass: Addison-Wesley Pub. Co., 1989. ISBN: 0201157675.

[Hak64] S. L. Hakimi. “Optimum Locations of Switching Centers and the Absolute Centers and Medians of a Graph”. In: *Operations Research* 12.3 (1964), pp. 450–459. ISSN: 0030-364X.

[Hak65] S. L. Hakimi. “Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems”. In: *Operations Research* 13.3 (1965), pp. 462–475. ISSN: 0030-364X.

[Hol93] J. H. Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. 2nd. pr. Cambridge (Mass.): The MIT Press, 1993. ISBN: 9780262581110.

[KD05] A. Klose and A. Drexl. “Facility location models for distribution system design”. In: *European Journal of Operational Research* 162.1 (2005), pp. 4–29. ISSN: 03772217.

[KK62] H. W. Kulin and R. E. Kuenne. “An efficient algorithm for the numerical solution of the generalized Weber problem in spatial economics”. In: *Journal of Regional Science* 4.2 (1962), pp. 21–33. ISSN: 0022-4146.

[Krane] A. Kransdorff. “The management page: High stock levels - Not the answer - Arnold Kransdorff reports on ’Supply Chain Management’”. In: *Financial Times* (June 4th 1982).

[LG05] K. S. Lee and Z. W. Geem. “A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice”. In: *Computer Methods in Applied Mechanics and Engineering* 194.36-38 (2005), pp. 3902–3933. ISSN: 00457825.

[LGR09] J.-E. Lee, M. Gen, and K.-G. Rhee. “Network model and optimization of reverse logistics by hybrid genetic algorithm”. In: *Computers & Industrial Engineering* 56.3 (2009), pp. 951–964. ISSN: 03608352.

[LH03] R. Lougee-Heimer. “The Common Optimization INterface for Operations Research: Promoting open-source software in the operations research community”. In: *IBM Journal of Research and Development* 47.1 (2003), pp. 57–66. ISSN: 0018-8646.

[LMW88] R. F. Love, J. G. Morris, and G. O. Wesolowsky. *Facilities location: Models & methods*. Vol. v. 7. Publications in operations research series. New York: North-Holland, 1988. ISBN: 0444010319.

[Mie58] W. Miehle. “Link-Length Minimization in Networks”. In: *Operations Research* 6.2 (1958), pp. 232–243. ISSN: 0030-364X.

[MJKSK06] H. Min, H. Jeung Ko, and C. Seong Ko. “A genetic algorithm approach to developing the multi-echelon reverse logistics network for product returns”. In: *Omega* 34.1 (2006), pp. 56–69. ISSN: 03050483.

[MND14] S. A. Melnyk, R. Narasimhan, and H. A. DeCampos. “Supply chain design: issues, challenges, frameworks and solutions”. In: *International Journal of Production Research* 52.7 (2014), pp. 1887–1896. ISSN: 0020-7543.

[MNG09] M. T. Melo, S. Nickel, and F. Saldanha-da Gama. “Facility location and supply chain management – A review”. In: *European Journal of Operational Research* 196.2 (2009), pp. 401–412. ISSN: 03772217.

[MSQ07] Z.-J. Max Shen and L. Qi. “Incorporating inventory and routing costs in strategic location models”. In: *European Journal of Operational Research* 179.2 (2007), pp. 372–389. ISSN: 03772217.

[OCD08] L. Ozsen, C. R. Coullard, and M. S. Daskin. “Capacitated warehouse location model with risk pooling”. In: *Naval Research Logistics* 55.4 (2008), pp. 295–312. ISSN: 0894069X.

[RS00] S.-u. Rahman and D. K. Smith. “Use of location-allocation models in health service development planning in developing nations”. In: *European Journal of Operational Research* 123.3 (2000), pp. 437–452. ISSN: 03772217.

[SCD03] Z.-J. M. Shen, C. Coullard, and M. S. Daskin. “A Joint Location-Inventory Model”. In: *Transportation Science* 37.1 (2003), pp. 40–55. ISSN: 0041-1655.

[She07] Z.-J. M. Shen. “Integrated supply chain design models: a survey and future research directions”. In: *Journal of Industrial and Management Optimization* 3.1 (2007), pp. 1–27. ISSN: 1547-5816.

[SLKSL04] D. Simchi-Levi, P. Kaminsky, and E. Simchi-Levi. *Managing the supply chain: The definitive guide for the business professional / David Simchi-Levi, Philip Kaminsky and Edith Simchi-Levi.* New York and London: McGraw-Hill, 2004. ISBN: 978-0071410311.

[SOU09] K. Sourirajan, L. Ozsen, and R. Uzsoy. "A genetic algorithm for a single product network design model with lead time and safety stock considerations". In: *European Journal of Operational Research* 197.2 (2009), pp. 599–608. ISSN: 03772217.

[SR02] Srinivas Talluri and R.C. Baker. "A multi-phase mathematical programming approach for effective supply chain design". In: *European Journal of Operational Research* 141.3 (2002), pp. 544–558. ISSN: 03772217.

[Sri95] R. Sridharan. "The capacitated plant location problem". In: *European Journal of Operational Research* 87.2 (1995), pp. 203–213. ISSN: 03772217.

[Ste89] G. C. Stevens. "Integrating the supply chain". In: *International Journal of Physical Distribution & Logistics Management* 19.8 (1989), pp. 3–8.

[STS05] J. Shu, C.-P. Teo, and Z.-J. M. Shen. "Stochastic Transportation-Inventory Network Design Problem". In: *Operations Research* 53.1 (2005), pp. 48–60. ISSN: 0030-364X.

[TG96] D. J. Thomas and P. M. Griffin. "Coordinated supply chain management". In: *European Journal of Operational Research* 94.1 (1996), pp. 1–15. ISSN: 03772217.

[Tho06] U. Thonemann. *Operations Management: [Konzepte, Methoden und Anwendungen].* [Nachdr.] Wi - Wirtschaft. München [u.a.]: Pearson Studium, 2006. ISBN: 3-8273-7120-1.

[Tiw+10] M. K. Tiwari, N. Raghavendra, S. Agrawal, and S. K. Goyal. "A Hybrid Taguchi–Immune approach to optimize an integrated supply chain design problem with multiple shipping". In: *European Journal of Operational Research* 203.1 (2010), pp. 95–106. ISSN: 03772217.

[Vaz02] A. Vazsonyi. *Which door has the Cadillac: Adventures of a real-life mathematician.* New York: Writers Club Press, 2002. ISBN: 0595260624.

[WP09] A. Weber and G. Pick. *Reine Theorie des Standorts.* Reine Theorie des Standorts. J.C.B. Mohr (Paul Siebeck), 1909.

[Yao+10] Z. Yao, L. H. Lee, W. Jaruphongsa, V. Tan, and C. F. Hui. "Multi-source facility location–allocation and inventory problem". In: *European Journal of Operational Research* 207.2 (2010), pp. 750–762. ISSN: 03772217.

[Enc] Encyclopedia of Mathematics. *Encyclopedia of Mathematics: Fermat-Torricelli problem.*

[End37] Endre Weiszfeld. “Sur le point pour lequel la somme des distances de n points donnes est minimum”. In: *Tohoku Mathematical Journal* (1937).

[Lau14] Laurent Perron. *Google OR-Tools*. 2014.

[ŞS07] G. Şahin and H. Süral. “A review of hierarchical facility location models”. In: *Computers & Operations Research* 34.8 (2007), pp. 2310–2331. ISSN: 03050548.