universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Novel Matheuristic for Two Echelon Inventory Routing Problems in lower-middle income countries“

verfasst von / submitted by

Rebekka Prader BSc MSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2023 / Vienna, 2023

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 066 977
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Masterstudium Business Analytics
Betreut von / Supervisor:	Univ.-Prof. Mag. Dr. Karl Franz Dörner, Privatdoz.

Abstract

This master thesis extends the inventory routing problem for a perishable good on two levels from Prader (2022) by adding an ALNS-algorithm that improves the assignment of customers to distribution centers. This step is especially important for supply chains in lower-middle income countries. Here, the street quality might be poor or subject to change, which can slow down or limit food delivery. Additionally, the vehicles used for transportation might not allow for sufficient cooling of food items or they might not protect them sufficiently from dust, smoke or other externalities. This can lead to a faster degradation of the perishable product, which might decrease food quality or safety.

As in Prader, we optimize storage and transportation decisions simultaneously and the age of the perishable item is tracked exactly. The objective function includes the reduction of food waste occurring during storage and transportation as well as keeping the (environmental) costs of the supply chain as low as possible. The solution method from Prader (2022) is improved and accelerated and the new ALNS to improve the customer assignment is tested extensively on several newly generated instance sets. Those new instances include different circumstances that are characteristic of lower-middle income countries and thereby help to demonstrate how broad the field of application of the solution method is.

Kurzfassung

Diese Masterarbeit erweitert das Inventory Routing Problem für ein verderbliches Gut auf zwei Ebenen aus Prader (2022) um einen ALNS-Algorithmus, der die Zuordnung von Kunden zu Zwischenlagern verbessert. Dieser Schritt ist besonders für Lieferketten in Ländern mit niedrigem bis mittlerem Einkommen essenziell. Das liegt zum einen daran, dass die Qualität der Straßen in solchen Ländern gering und wechselhaft sein kann, was den Transport hier stark verlangsamen und einschränken kann. Zudem kommen hier oft Fahrzeuge zum Einsatz, die Lebensmittel nicht ausreichend kühlen oder vor Externalitäten schützen. Das kann dazu führen, dass der Verderb von Lebensmitteln auf der zweiten Stufe beschleunigt wird oder die Qualität und Sicherheit der Nahrungsmittel beeinträchtigt wird.

Wie in Prader werden Lagerentscheidungen sowie Lieferrouten optimiert und das Alter des verderblichen Gutes in der Lieferkette wird exakt bestimmt. Die Reduktion von Lebensmittelabfällen bei der Lagerung, Transport- und Lagerkosten sowie der Verderb auf der zweiten Stufe werden simulatan in der Zielfunktion betrachtet. Der Lösungsansatz aus Prader (2022) wird verbessert und beschleunigt und der neue ALNS zur Veränderung der Kundenzuordnung wird anhand von neuen Instanzensets ausgiebig getestet. Die neuen Instanzen beinhalten verschiedene Eigenschaften von Lieferketten in Ländern mit niedrigem bis mittlerem Einkommen, um die breite Einsatzmöglichkeit der Lösungsmethode zu veranschaulichen.

Contents

Abstract i

Kurzfassung iii

List of Abbreviations vii

List of Tables ix

List of Figures xi

1 Introduction 1

2 Literature Review 3
- 2.1 Literature related to IRPs for perishable items . . . 3
- 2.2 Literature related to 2E-IRPs for perishable goods . . . 4
- 2.3 Food Security Index . . . 5
- 2.4 Income levels of the World Bank . . . 6
- 2.5 Contribution . . . 7

3 Problem Formulation 9
- 3.1 Summary of the original formulation . . . 9
  - 3.1.1 Delivery patterns in the original formulation . . . 9
  - 3.1.2 Deterioration in the original formulation . . . 10
  - 3.1.3 Objectives in the original formulation . . . 11
- 3.2 Adaptations of the formulation . . . 11

4 Summary of the Mathematical Model from Prader 13
- 4.1 Decision Variables . . . 13
- 4.2 Sets and Parameters . . . 14
- 4.3 Objective Function from Prader . . . 14
- 4.4 Model Constraints from Prader . . . 16

5 Solution Method 17
- 5.1 Summary of the original solution method . . . 17
- 5.2 Summary of the three ALNS algorithms . . . 18
  - 5.2.1 The ALNS-2LR . . . 19
  - 5.2.2 The ALNS-DP . . . 22
  - 5.2.3 The ALNS-CCA . . . 24

6 Instance Generation and Model Adaptations **29**
6.1 Instance Set 1: Urban food delivery 31
6.2 Instance Set 2: Lower-middle income level countries 31
6.3 Instance Set 3: Delivery of villages 32
6.4 Instance Set 4: Changing street quality 32
6.5 Instance Set 5: Disaster zone delivery 33

7 Computational Study **35**
7.1 Instance Set 1 and general parameter setting for the ALNS-CCA 37
7.1.1 Initial clustering 37
7.1.2 Stopping criterion for the first level 38
7.1.3 Stopping criterion for the second level 38
7.1.4 General Parameter setting on Instance Set 1 38
7.2 Instance Set 2 46
7.3 Instance Set 3 48
7.4 Instance Set 4 51
7.5 Instance Set 5 54
7.6 Efficient combination of ALNS-CCA and ALNS-DP 56
7.6.1 Improvements and tests of the ALNS-DP 57
7.6.2 Conclusion on the ALNS-CCA and ALNS-DP 58

8 Conclusion **59**

9 References **61**

List of Abbreviations

ALNS	adaptive large neighbourhood search
ALNS-CCA	adaptive large neighbourhood search to change customer assignments
ALNS-DP	adaptive large neighbourhood search to change delivery patterns
ALNS-2LR	adaptive large neighbourhood search to change 2nd level routing
CTD	close to deterioration
CVRP	capacitated vehicle routing problem
DC	distribution center
GFSI	Global Food Security Index
GHG	greenhouse-gases
GNI	Gross National Income
IRP	inventory routing problem
MILP	mixed integer linear program
PIRP	perishable inventory routing problem
UN	United Nations
UNCCD	United Nations Convention to Combat Desertification
USD	US dollar
VRP	vehicle routing problem
WHO	World Health Organization
2E	two echelon

List of Tables

3.1 Reprinted from „Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective“, by Prader, R., 2022., p.8. 10

4.1 overview of the relevant decision variables of the mathematical model adapted from „Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective“, by Prader, R., 2022., p.12. 13
4.2 overview of all relevant sets and parameters used in the mathematical model adapted from „Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective“, by Prader, R., 2022., p.11. 14
4.3 overview of the factors in the consumption model reprinted from „Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective“, by Prader, R., 2022., p.13. 15

6.1 8 Versions tested for most instance sets 30

7.1 Improvement on both levels found by 50 iterations of the ALNS-CCA. The performance of the algorithm with 10 iterations of the ALNS-2LR to generate the second level solution is tested by comparing the improvement found like this to the "real" solution. "Real" here means the solution with 1000 iterations of the ALNS-2LR for large instances and the optimal solution with gurobipy stopped at five percent from the optimal for the small instances. 39
7.2 Improvement on both levels found by 50 iterations of the ALNS-CCA, starting from a distance-based customer assignment. The performance of the algorithm with 10 iterations of the ALNS-2LR is compared to the "real" solution. 41
7.3 Improvement on both levels found by 50 iterations of the ALNS-CCA, starting from a regret based customer assignment. 41
7.4 Average cost improvement found and runtime of the ALNS-CCA on eight instances over five runs. For each instances, 5, 10 or 20 iterations of the ALNS-CCA without an improvement are used as a stopping criterion. . . 42
7.5 Number of improvements found by different operators and operator combinations in the runs from table 7.4 with 20 iterations as a stopping criterion for instances N400-Gs-Tr-U, N400-Gl-Tr-U and N400-Gl-Tbl-U and for 5, 10 and 20 iterations for instance N400-Gs-Tbl-U. 45

7.6 Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 2. The starting assignment is created by the distance based algorithm from 7.1.1. 47
7.7 Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 2. The starting assignment is created by the random algorithm. 47
7.8 Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 2. The starting assignment is created by the regret based algorithm. . 48
7.9 Improvement found over 5 runs by the ALNS-CCA with varying numbers and sizes of villages. 49
7.10 Total costs of the best found solution with the original costs (total costs before) compared to the costs of that solution after a cost change (total costs after). The last column of the table presents the total costs of a solution with the new costs that is again improved by the ALNS-CCA (total costs ALNS-CCA). 52
7.11 Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 5. The starting assignment is created by the distance based algorithm from 7.1.1. 55
7.12 Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 5. The starting assignment is created by the random customer assignment algorithm. 56
7.13 Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 5. The starting assignment is created by the regret customer assignment algorithm. 56
7.14 Results of a run with the following steps: first the ALNS-CCA is used on the starting solution. The found total costs are reported (c after ALNS-CCA) and the found customer assignment is passed on to the ALNS-DP that additionally improves the delivery patterns with 100 iterations as a stopping criterion. The costs of the best found solution here are again reported (c after ALNS-DP), together with the number of improvements found by the feedback loop (# imp.) and the runtime of the ALNS-DP (rt. ALNS-DP) with the new settings. 57

List of Figures

7.1 Random starting assignment (left) and improved customer assignment after the application of the ALNS-CCA (right). 43
7.2 A typical instance of instance set 2 after the application of the ALNS-CCA. 48
7.3 One of the found final customer assignments on version 1 from instance set 3. 50
7.4 One of the found final customer assignments on version 2 from instance set 3. 51
7.5 One of the found final customer assignments on version 3 from instance set 3. 51
7.6 One of the found final customer assignments on version 4 of instance set 4. 53
7.7 One of the found final customer assignments on the small version of instance set 4 with the terminal at the bottom left of the grid. 53
7.8 One of the found final customer assignments on the large version of instance set 5 with the terminal at the bottom left of the grid. 54
7.9 One of the found final customer assignments on the small version of instance set 5 with the terminal at a random position. 55

1 Introduction

Supply chains for perishable food items exist all over the world and are adapted to very diverse conditions. While street quality, storage practices, means of transportation, speed and mode of deterioration can differ strongly in different parts of the world, there are two important things that always need to be considered with this type of product: waste reduction and food safety. The first is a critical issue in a world where hunger is generally rising since almost a decade while 14 percent of food production, according to the United Nations, end as waste post-harvest and before retail. Taking action to reduce these numbers is also important for environmental considerations, since horticultural products are produced, stored and transported without in the end being consumed by the final customer. For instance, 38 percent of the energy in the global food system are used on wasted food. Additionally, all these steps, including waste disposal itself, are also connected to the emission of greenhouse gasses, which considerably undermines the sustainability of global food provision. (United Nations, 2022)

The second important issue to address is food safety, which according to the WHO Global Strategy for Food Safety 2022-2030 is about guaranteeing that consumers all over the world have access to "safer food for better health". (World Health Organization, 2022) How different the status quo of food safety is in different countries is illustrated by the Food Security Index developed by Economist Impact for the United Nations. It ranks 113 countries based on 4 factors: affordability, availability, quality and safety and, finally, sustainability and adaptation. While the present work will not deal with the availability aspect, all others will be addressed to different extends in the formulation of the problem and the generation of test instances.

Generally speaking, the present master thesis is an extension of Prader (2022). This original work presented a two echelon inventory routing problem (2E-IRP) for perishable horticultural goods and included waste considerations into the objective function. However, the model presented by Prader is designed specifically for an urban context where fresh produce is harvested in the outskirts of a city like the Austrian capital Vienna and delivered to customers within the city. If this model shall be applied to different types of two echelon supply chains, however, some of the made assumptions might become disadvantageous. The goal of the present work is to extend the solution method from Prader in a way to make it more applicable to diverse food supply chains in different parts of the world. Adaptations of the model and of the solution method suited for the considered supply chain types will be created and shall then be tested extensively on more diverse test instances. The newly generated instances try to cover, without fully exhausting them of course, diverse supply chain conditions and objectives regarding food waste reduction and food safety. The main focus will be on lower-middle income countries based on the classification of the world bank. Some of the considered circumstances are

based on our observations of street and transportation conditions in Nepal, Lao, Vietnam and Indonesia.

One main methodological adaption will be the development of a new adaptive large neighborhood search (ALNS) procedure to change customer assignments. Those were assumed to be fixed in the original work which might become disadvantageous in supply chains in some parts of the world. The newly generated ALNS, that will be referred to as ALNS to change customer assignments, short ALNS-CCA, also proved to be an efficient method to generate a customer clustering for transportation. Other contributions of this work are the improvement and more profound testing of the original method as well as the generation of several new test instances representing food supply chains in different parts of the world.

This master thesis will be structured into the following chapters. First, a literature review on related topics and a more detailed description of the food security index as well as the situation in different countries, will be given. This will be followed by a description of the original problem and its adaptations, including the new ALNS procedure. After that new test instances will be generated taking into consideration different global supply chain conditions and focusing on aspects regarding food waste and safety. After a parameter setting for the ALNS-CCA a detailed computational study will be performed in which the new method will be tested. The computational study also aims at finding improvements of the original method, especially by speeding up the feedback loop presented in Prader (2022).

2 Literature Review

The Inventory Routing Problem (IRP) is a special case of the Vehicle Routing Problem (VRP) introduced by Dantzig and Ramser in 1959. While the general VRP solves a problem in which a set of cost-minimizing vehicle routes from a depot is generated to fulfill all demand of the given customers, the IRP additionally aims at optimizing the inventory quantities at given storage locations. This means, in such a problem routing and inventory decisions are taken simultaneously. Based on the definition by Coelho et al. (2014) in an IRP the supplier decides about the timing, quantity and routes for customer delivery.

Since in Prader (2022) the literature on storage and distribution of perishable items in two echelon problems is already analyzed in much detail, we want to refer to this work for a general overview. The focus of the following literature review will be, after a short overview over the most relevant literature consulted in Prader, on the newly included aspects as food safety in diverse global supply chains.

2.1 Literature related to IRPs for perishable items

As mentioned in Prader, the reduction of food wastage in supply systems can be achieved by improving production or location decisions as well as by optimizing inventory and transportation. While Prader focused on storage and transportation, the present work for some cases additionally includes location decisions to a certain extend. To be precise, even though the locations of storage facilities on both levels are considered as fixed and given, the customer assignment in some instances are not fixed anymore. Like this some transportation distances change which, in the best case, helps reducing route length and like that deterioration on the second level.

In this context it is important to mention the work by Rabbani et al. (2016), who present a genetic algorithm to solve a single-period problem with several intermediate depots. While Prader (2022) track the precise age of transported and stored goods, in Rabbani the deterioration only depends on the time a good spends on the vehicle used for transportation. The present master thesis will apply a combined version, where the age in days of products in the supply chain in tracked to determine the amount of food that is wasted during storage. What will be added is the consideration of the transportation duration on the second level, where deterioration is assumed to happen faster compared to storage and first level transportation. In the considered supply chains this can have different reasons, starting from the low transportation speed of second level vehicles, bad street conditions for last mile delivery or inefficient cooling in the used vehicles. Like this, the delivery duration on this echelon will increase the chance that the quality and safety

of the good arriving to the final customer is actually much worse than if it was only stored in optimal conditions up to this point. This, finally, increases the chance of wastage after delivery or the consumption of unsafe food items. Both types of deterioration are relevant in the case of perishable food items since storage as well as delivery times are, in different ways, relevant for the quality of the goods arriving at the consumer.

While Rabbani et al. do not take inventory decisions into considerations, there has also been a vast amount of research on IRPs for perishable items. One of the first to consider such a case, where routing and inventory of a perishable good are managed in combination, is the work by Federgruen et al. (1986). Later, Hiassat and Diabat (2011) developed a similar model where a single perishable product is delivered from a central depot, over several distribution centres (DCs), to a set of retailers. Here a decision on which DCs to open and which customers to assign to them, is made. Hiassat et al. (2017) finally solved such an integrated location-inventory-routing problem by a genetic algorithm. However, waste reduction and food safety were not taken into consideration explicitly and routes were just picked out of a set of given feasible routes. Other peers that developed genetic algorithms to solve similar IRPs are Azadeh et al. (2017) and Rahimi et al. (2017).

One work that is of particular relevance for the development of the model in Prader (2022) is the one of Soysal et al. (2015). Both works optimize IRPs with costs for routing, waste, and fuel consumption in the objective function. However, Soysal deals with a stochastic problem for a perishable good. In this work a method based on simulation is used to solve the developed chance constrained problem. However, this paper only considers a single echelon and does not present a deterministic solution approach.

Some other papers that are strongly related to the model from Prader (2022) are dealing with agri-food supply chains for perishable goods. Examples here are the works of Onggo et al. (2019), which deals with a periodic IRP with stochastic demand, and Violi et al. (2020), who present a stochastic IRP of a perishable product on multiple stages. All of the papers mentioned so far, however, do not consider two echelon transportation systems.

2.2 Literature related to 2E-IRPs for perishable goods

While 2E-VRPs have been studied extensively in the past, 2E-IRPs are a relatively new field. For the first mentioned, as Prader (2022), we want to refer to the review by Cuda et al. (2015) for an overview.

The only one that needs to be mentioned here is Hemmelmayr et al. (2012) since it was the basis of all ALNS procedures presented in Prader (2022) and its concepts will also be used to generate the new ALNS in the present master thesis. Hemmelmayr et al. designed an ALNS for a 2E-VRP which works with two classes of destroy operators, some with a small and some with a large impact. The first mentioned destroy only the second level routes while the large impact operators cause changes that influence the complete solution.

The papers published on 2E-IRPs so far are, as already mentioned, not as numerous as those dealing with 2E-VRPs. One example is the Zhao et al. (2008), which introduces an IRP on three echelons and solves it by a variable large neighbourhood search heuristic.

However, perishability is not taken into consideration here. Rohmer et al. (2019), present a 2E-IRP for perishable items with a single supplier delivering a single DC. The supplier in this model picks the delivery patterns for the final customers and the transportation routes are optimized. The proposed solution method, a matheuristic, is the basis of the one presented in Prader (2022). It consists of the optimal solution of a reduced mathematical formulation which excludes second level routes, and an ALNS procedure to generate this missing element in a second step. Some main contributions of Prader, compared to Rohmer et al., were integrating more than one intermediate distribution center as well as dealing with a more complex objective function including waste considerations.

Guimarães et al. (2019) developed a 2E-IRP where the DCs are responsible for the organisation of pickups from the suppliers as well as deliveries to the customers. The problem is optimally solved by a branch and cut algorithm while a rich matheuristic solves the problem for large instances. Farias et al. (2020) formulate a 2E-IRP mathematically for several inventory policies. Even though their model is of higher complexity than many of the previously mentioned ones, no solution method to solve large problems is presented. Farias, as Prader (2022) but unlike the present work, also considers a fixed customer assignment to DCs. Schenekemberg et al. (2020) deal with an 2E-IRP with fleet management considerations. Schenekemberg et al. (2021) formulate a 2E production-routing problem that they solve with a branch and cut algorithm. As mentioned at the beginning of this literature review, the production is not considered in the present work. Finally, Charaf et al. (2022) present a branch-and-price algorithm to solve a route-based formulation of the 2E-IRP and test it on 400 new test instances.

2.3 Food Security Index

The Global Food Security Index (GFSI) developed by Economist Impact scores 113 countries based on 28 indicators in four categories: affordability, availability, quality and safety as well as sustainability and adaptation. Food security is, on the website of the UN, defined as "the state in which people at all times have physical, social and economic access to sufficient and nutritious food that meets their dietary needs for a healthy and active life". (UNCCD, 2022)

The present master thesis, as stated in the introduction, tries to include three of the four categories into the objective, while availability is not addressed.

Affordability shall be guaranteed by keeping the costs of the supply chain as low as possible, including costs of ordering, storage, delivery and food wastage. Guaranteeing high food quality is included in several ways. On the one hand, as in Prader (2022) and Rohmer et al. (2019), age-dependent holding costs make sure that the age of goods delivered to the final customer is as low as possible. On the other hand, additionally to the considerations in Prader, the current work tries to keep second level routes as short as possible by improving the customer assignment. Especially if transportation conditions on the second level enhance deterioration, this is an important step to guarantee high food quality and safety at a consumer level. Food safety is also guaranteed in the model by not allowing for the delivery of deteriorated food to the final customer. Last but not

least, sustainability and adaptation were already the main objective in Prader (2022) since the objective function focuses on the reduction of greenhouse gas emission and food waste. The present work, as stated previously, tries to improve this environmental impact even further.

While the new instances try to include many aspects that might become relevant, especially in lower-middle income countries, we want to emphasize again that it is of course impossible to cover all possible characteristics of the diverse supply chains all over the world. However, our aim is to generate a test set that explores realistic circumstances in countries on the second income level based on the classification of the World Bank, that will be introduced in more detail in the next section.

The four countries that we observed during the research for this thesis, namely Indonesia, Lao, Nepal and Vietnam, score differently on the GFSI. While Indonesia performs quite well in affordability, here especially the sustainability aspect must be improved. The country currently ranks 83rd in this category, followed by Nepal on place 83. Nepal is only on the 72nd place for quality and safety due to the difficult transportation conditions. Lao is around 80st of the tested countries in all categories. And finally, Vietnam is doing relatively well in most categories, however, the environmental impact is relatively high in this country as well.

2.4 Income levels of the World Bank

As already mentioned, the focus of the novel instances will be on lower-middle income countries and our observation of their transportation conditions in Nepal, Lao, Vietnam and Indonesia. The World Bank classifies all countries in the world into four different income levels that define life standards and, among others, access to sufficient and safe food. Each year on the first of July the classification, that is based on the GNI of the previous year, is reviewed. While European countries like Austria, that was considered in the original work of Prader, are on the highest income level, street conditions in poorer countries are often worse which can lead to higher costs and a lower resilience of the supply system.

The lower-middle income level is the second lowest out of four income levels defined by the Swedish researcher and professor of international health at Karolinska Institute Hans Rosling. His system was adopted by the World Bank as a representation of living conditions in different parts of the world. According to Roslings website gapminder nearly half the world's population lives on this income level. Generally, people in countries on this level earn between two and eight USD a day and their basic needs are met. The majority of the population in lower-middle income countries are farmers or work in other physical jobs. However, unlike people on the lowest income level they are able to save some money which enables them to buy some food items as meat, eggs or vegetables they do not produce themselves. Most people on this income level can afford to cook on gas stoves and have some, even though eventually unstable, electricity that can help to keep food fresh for longer.

Countries on the fourth and highest income level were dealt with in the original work of

Prader (2022) and will also be considered in the first instance set of the present work. The transportation conditions in countries on the third level might lay somewhere between those and the ones on level two.

However, we will not deal with low income countries. One main reason is, that people living here, earning less than two USD per day on average, do not have the financial means for buying foods they do not produce themselves. For this reason, food supply chains as the ones considered in this work might not be relevant in such regions of the world.

2.5 Contribution

Prader (2022) filled a research gap by developing a complex 2E-IRP for a perishable item and an efficient solution method for this problem. The present thesis works on the same problem as Prader but adds some elements to the solution method and tests it on more diverse instances. To be precise, the new aspects in the present work are the following:

- A second ALNS algorithm is added which changes the assignment of customers to DCs, which was fixed in the original method. This is expected to make the second level routes shorter which is especially important if fast deterioration happens in this step. It also helps to react quickly to changing road conditions, as the computational study will show.
- The feedback loop from Prader (2022) is accelerated by making the improvement method from the original method faster and more efficient.
- For small instances of 50 customers and 3 DCs we found a way to solve the complete problem optimally for a case with given delivery patterns and given customer assignments. This helps to evaluate the real impact of the two presented improvement algorithms and works by deconstructing the problem and solving each level separately with the gurobipy solver.
- Several novel sets of instances representing different circumstances of global supply chains are introduced. The focus here lies on the representation of conditions in lower-middle income countries.
- The new improvement method, the ALNS to change customer assignment, is tested extensively on the novel instance sets.
- Finally, it is tested how the two improvement methods, the one from Prader (2022) which changes delivery patterns and the new one, changing customer assignments, work together if used consecutively.

In general, the present master thesis develops a complete solution method with two different improvement algorithms that can be used in combination or independently. Improving the solution as much as possible helps to reduce the environmental impact of

the supply chain as well as the amount of waste occurring during all steps, transportation as well as storage. Accelerating transportation in this case also helps to improve food security in its different aspects.

3 Problem Formulation

3.1 Summary of the original formulation

In Prader (2022) a new two echelon inventory routing problem with waste related objectives was designed for an urban transportation case and solved by a matheuristic including two different ALNS algorithms. It was tested mainly on instances where a central depot (or central terminal) outside of the city center is used for storing a perishable horticultural good after it is harvested. On the first level, large combustion engine vehicles deliver this good to several distribution centers (DCs) that are spread over the city center. The second level consists of the delivery of final customers from the DCs by cargo bikes. Prader (2022) assumes that the assignment of customers to DCs is fixed, which means that independently of demand quantities and other externalities, a DC always delivers the same set of customers. The decision maker can choose when to deliver final customers from a set of acceptable delivery days. This leads to a case where the delivery patterns are the only connection between the first and the second level, which makes it possible to decompose the problem and solve the two levels separately. In the original paper the solution of the first level, including the first level routes, inventory quantities at the DCs and the delivery patterns, is found by optimally solving a reduced version of the mathematical problem formulation. The second level routes, on the other hand, are solved for each of the DCs separately by generating an initial solution using a cheapest insertion algorithm and improving it by an ALNS. The resulting complete solution can, finally, be improved by finding better delivery patterns using a feedback loop which includes a second ALNS algorithm.

In the original formulation the storage capacity of the DCs is limited but the central terminal has infinite capacity. In the urban transportation case of Prader, consolidation at the DC is mandatory, either for environmental reasons or because of traffic limitations within the city. The present work also uses this assumption but adds an additional reason for changing the vehicles on the second level: street conditions that do not allow for big trucks to drive.

3.1.1 Delivery patterns in the original formulation

As already mentioned, the delivery periods for each customer can be chosen by the supplier in the model of Prader (2022). Like this the amount of waste occurring during storage can be reduced since it allows the supplier to fulfil the demand of some customers in moments where a lot of close-to-deterioration (CTD) goods are on stock. Of course, the decision on the delivery days for final customers has a strong influence on the stored quantity needed in different moments in time. This changes the routing costs on both

levels, since goods need to be delivered in the right moment. It also impacts storage costs since the storage time of goods is altered by this decision.

The example in 3.1 comes from Prader (2022) and represents a small problem with five customers, A to E. For each of them the number of days per week that they want to be delivered on is given in brackets. In the example this is either one day or two. The grey fields in the table are days on which customers are unavailable while the delivery company is allowed to pick the delivery dates between the white fields, which are the days on which a customer can be visited. The example, as Prader (2022) and the current work, assumes a planning period of five days. The black dots represent the finally chosen delivery days.

Customer	Day 1	Day 2	Day 3	Day 4	Day 5
A (1)		•			
B (2)				•	•
C (1)		•			
D (2)			•		•
E (1)					•

Table 3.1: Reprinted from „Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective“, by Prader, R., 2022., p.8.

If, in this given solution, customer A and B are assigned to DC 1 and the remaining three are assigned to DC 2, only customer A is delivered by DC 1 on day 2 and only customer C by DC 2. No customer is delivered from DC 1 on day 3 and only customer D from DC 2. On the fifth day, DC 2 has two customers to deliver, namely D and E. The VRP that needs to be solved on the second level in this case only consists of finding the order in which these customers are delivered.

As this example shows, as soon as the customer assignment and the delivery patterns are given, the second level is completely deterministic and reduces to the solution of independent VRPs for each DC in each period. Since customer demands for each delivery are fixed and given as well, the total demand of a DC in each period is also known in such a case.

3.1.2 Deterioration in the original formulation

According to literature, there are two main ways of modeling deterioration. One option is using an expiry date, which requires tracking the age of a good precisely and allowing for delivery until this age is reached. This model, as described in Nahmias (2011), is often used for meat or dairy products. Prader (2022) models deterioration in this way to determine the amount of waste that occurs during storage. This means that after a certain number of periods a good spends in the supply chain, it is wasted and cannot be delivered anymore. The second type of deterioration modeling is common for many horticultural goods: gradual deterioration. Rohmer et al. (2019) employ this type of deterioration by integrating age-dependent holding costs into the model. Prader does

this as well but, as already mentioned, also penalizes goods that reach their end of life by including waste penalties into the objective functions. However, both of these works like this only penalize deterioration based on the age of the product while assuming, that the transportation occurs on a single day. That means that the length of the routes on both levels does not contribute to wastage differently than if goods were stored for the same amount of time. This might be realistic if the transportation does not take very long or if the used vehicles allow for efficient cooling. However, if this is not the case, transportation might cause faster degradation of a good.

For this reason, the present work will assume that the duration of second level routes, although no goods are wasted here directly, impacts the quality of food items arriving at the consumer. This gives an additional importance to making second level transportation as short as possible.

3.1.3 Objectives in the original formulation

The original formulation by Prader, as mentioned previously, is partially influenced by the work of Rohmer et al. (2019). Both models share the organization of the supply chain on two echelons, the use of age-dependent holding costs and that the inventory decisions made by the supplier are mainly about the selection of delivery patterns. Prader, however, adds some complexity to the model by including more than one DC. Since this requires considerable changes in the mathematical formulation, the work of Farias et al. (2020), that models such a 2E-IRP for multiple DCs, was an important influence of Prader for the development of the model. One of the main extensions compared to Farias is including waste and deterioration as well as applying a more complex objective function. Also, Prader, unlike Farias, present a heuristic solution method for the model. For the generation of the objective function, Soysal et al. (2015) delivered some important inputs.

The objective in Prader is to reduce total environmental costs of transportation, consisting of routing, storage and waste costs. The routing costs are assumed to be the sum of emission impact (based on the emission model from Ehmke et al. (2018)) and driver costs. Of course, if a vehicle does not emit greenhouse gasses, as it is the case for cargo bike for instance, only the driver wage is relevant. The model optimizes the delivery routes on both levels, finds the optimal stock level in each DC in each period and picks delivery patterns for each customer. The demand of the final consumers is considered as fixed and given. The same holds for the supplied quantity of the food item. However, if the supply is not sufficient, outsourcing is possible since the harvested quantity is not changeable in the short term and all demand needs to be satisfied.

3.2 Adaptations of the formulation

The present master thesis basically works on the same problem as Prader (2022) but improves the solution method. This is, later, tested on different new test instances.

One thing that all the new instances have in common is that the means of transportation on the second level are slower than on the first one and conservation of the perishable

good is more difficult. While this is already the case for the cargo bikes used in the original model, in other countries the situation might be even more extreme. In Nepal, for instance, last mile delivery is often performed by mules or human carriers where the packaging of the goods is their only protection from the sun, dust and other external factors. Additionally, the delivery speed is very low and some of the used streets are hardly walkable. This might be an extreme example but it shows clearly, that especially last mile delivery conditions can be very divers in different countries. This makes the optimization of the duration of second level routes crucial to reduce food wastage and enhance food safety.

Another thing that will be considered in some of the test instances is that the transportation speed on the second level might vary strongly depending on the taken road because of street conditions. Especially if street conditions are subject to change, as in areas with frequent landslides, floods or just even animals on the street, the assumption that the customer assignment to DCs is fixed can have a strong negative impact on the solution quality. Therefore, many of the new cases require getting rid of the assumption made by Prader that each DC always delivers a fixed set of customers, independently of the street quality, current demand and cost settings. Also, especially for customers with a similar distance to different DCs, this assumption might have a considerable negative effect already for the relatively standard case of urban transportation with cargo bikes. In a case where transportation duration on the second level is relevant for deterioration, such worse solutions might translate to higher levels of food waste or, for some developing countries, even the consumption of contaminated or unsafe food.

To sum up, the major methodological change compared to Prader (2022) is finding a way of making the customer assignment flexible without causing a considerable increase of complexity of the solution method. Therefore, the mathematical formulation generally only changes in a way that customer assignments are not fixed anymore. Some of the new instance sets include special assumptions that require some additional light adaptations of the formulation. Which changes those are will be explained in more detail in chapters 4 and 6.

4 Summary of the Mathematical Model from Prader

Prader (2022) formulated the new 2E-IRP as a mixed integer linear program (MILP) for a planning horizon of one week. To be precise, period 0 is a dummy period used to initialize the inventory and no supply or transportation happens. Operations start on day one and, in most cases, a total of five delivery days is assumed. Each good enters the supply chain at an age of 0 and gets one day older in every passing period. That means that the amount on stock in the morning of each day is the stored quantity of the previous day, after all delivery, aged by one. The maximum lifetime of a good is called m and stored goods with an age of m-1 are turned into waste immediately at the beginning of the next day. These goods are not considered inventory on this day anymore and deteriorated produce is never delivered to final customers.

As the model deals with inventory routing, both, transportation and stocking decisions are considered. As already mentioned, the supply quantity, for instance the amount of the perishable good harvested on a certain day, is considered as given and unchangeable in the short run. Missing produce can be outsourced to avoid stock-outs and the model determines this order quantity as a decision variable. In the original model by Prader the supplier is allowed to pick the delivery days. As Prader proved, the amount of waste occurring during storage depends on this decision to different extends based on the given cost settings.

4.1 Decision Variables

The following decision variables from Prader (2022) are necessary to summarize the model for our purpose. For the other used variables we would like to refer to the original work.

Variables	
$x_{i,j}^{k,t}$	1 if j is visited immediately after i by vehicle k $\in K$ in t $\in T$
$I_i^{t,g}$	inventory of age g $\in G$ at i $\in V_0 \cup V_{DC}$ at the end of period t $\in T \backslash \{0\}$
W_i^t	units of waste at node i $\in V_0 \cup V_{\mathrm{DC}}$ at the end of period t $\in T$
r_t	order quantity at terminal in period t $\in T$

Table 4.1: overview of the relevant decision variables of the mathematical model adapted from „Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective“, by Prader, R., 2022., p.12.

4.2 Sets and Parameters

The following table summarizes the sets and parameters from Prader (2022) that are relevant for the summary here.

Sets	
V	set of all nodes (terminal, DCs, customers), V = $V_0 \cup V_{DC} \cup V_C$
V_0	{0} = major terminal/central depot
V_{DC}	{1, ..., $\|V_{DC}\|$} = distribution centres (DCs)
V_C	{$\|V_{DC}\| + 1, \ldots, \|V\|$} = final customers , supplied only by the DCs (not V_0)
K_e	set of all vehicles that are available on levels e = {1,2}
K_1	{1, ..., b_1} = vehicles on the first level (departing from the terminal)
K_2	{1, ..., b_2} = vehicles on the second level (departing from the DCs)
T	{1, ..., t} = planning horizon
G	age of a product g = {0, 1, ..., m-1}, deteriorates at age m
Parameters	
c_t	order costs for externally procured products in period t $\in T$
o_e	wage rate of drivers on both levels e = {1,2} in €/s
p	penalty costs for wasted products (in €/unit)
m	fixed maximal shelf life of the product (at this age it is wasted)
h_i	holding costs/unit at each i $\in V$ per time period t $\in T \backslash \{0\}$
a_{ij}	distance between i & j for each i,j $\in V$
f_{ij}	average vehicle speed in m/s on each edge $i, j \in V_0 \cup V_{\mathrm{DC}}, \quad i \neq j$
l	fuel price/litre

Table 4.2: overview of all relevant sets and parameters used in the mathematical model adapted from „Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective“, by Prader, R., 2022., p.11.

4.3 Objective Function from Prader

The objective function in Prader (2022) consists of 6 different elements and aims at capturing the total (environmental) costs of transportation and storage.

Part (1) is an estimate of occurring fuel costs which is based on the S_FUEL model from Ehmke et al. (2018). Even though this model generally underestimates the true fuel consumption it is a relatively sophisticated measure of the environmental impact of the emitted greenhouse gasses. It is independent of the vehicle load which makes it easier to use since only the gross weight of the empty vehicle is needed. Average speed values are used for each arc. Table 4.1 from Prader, here reprinted in table 4.3, gives an overview of the factors that are used in the consumption model by Ehmke et al. (2018), which will stay the same in the present work. As in Prader, the original kN_ev, is summarized by the parameter y.

In Prader it is assumed that greenhouse gasses are emitted only on the first level. The present work will not keep this assumption for all cases so in some of them this type of cost will be relevant on all arcs, not only the ones on the first echelon.

$$\text{Minimise} \sum_{i,j \in V, i \neq j} \sum_{k \in K_1} \sum_{t \in T} \lambda (y \left(\frac{a_{i,j}}{f_{i,j}} \right) x_{i,j}^{k,t} + \gamma \beta a_{i,j} f_{i,j}^2 x_{i,j}^{k,t} + \gamma \alpha \mu x_{i,j}^{k,t} a_{i,j}) \tag{1}$$

$$+ \sum_{e \in \{1,2\}} \sum_{i,j \in V, i \neq j} \sum_{k \in K_e} \sum_{t \in T} \left(\frac{a_{i,j}}{f_{i,j}} \right) x_{i,j}^{k,t} o_e \tag{2}$$

$$+ \sum_{t \in T} r_t c_t \tag{3}$$

$$+ \sum_{g \in G} \sum_{t \in T} \sum_{i \in V_0 \cup V_{\text{DC}}} h_i^g I_i^{t,g} \tag{4}$$

$$+ \sum_{t \in T} \sum_{i \in V_0 \cup V_{\text{DC}}} W_i^t p \tag{5}$$

$$+ \sum_{i \in V_0 \cup V_{DC}} I_i^{T,m-1} p \tag{6}$$

Notation	Description	Value
λ	product of different factors from Franceschetti et al. (2013)	1/32428
k	engine friction factor	0.2
N_e	engine speed	33
v	engine displacement	5
γ	product of different factors from Franceschetti et al. (2013)	0.0028
β	product of different factors from Franceschetti et al. (2013)	1.6487
α	product of different factors from Franceschetti et al. (2013)	0.0981
μ	curb-weight	12700
l	fuel price per litre	1.8

Table 4.3: overview of the factors in the consumption model reprinted from „Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective“, by Prader, R., 2022., p.13.

(2) estimates the driver costs which in this case are only dependent on the costs of an edge (so the delivery time) on all levels. Part (3) considers outsourced quantity and integrates the resulting order costs for externally procured products into the objective function. The present work, as Prader (2022), assumes that orders can only be made at the major terminal, which is why the order variable only has the respective period in the planning horizon as an index. The age-dependent holding costs of the major terminal

and the DCs are captured by part (4) of the objective function. Part (5) and (6) add the waste costs caused by too long storage of the perishable good. The second mentioned is required since goods that have an age of m-1 on the last day of the planning horizon will inevitably be wasted on the next morning. This would not be considered if only (5) was included.

The adapted versions of the model will, for instance, include higher transportation costs on some edges because of the fast deterioration happening here. However, to simplify this we do not change the objective function directly but mainly increase the cost or length of an edge which in the formulation is represented by the parameter a_{ij}.

4.4 Model Constraints from Prader

The complete mathematical formulation of Prader, even though most constraints stay the same for the present work, will not be reprinted here. Instead, the following section will give an overview of the constraint classes used in the original model.

The first set of constraints in Prader (2022) deals with the inventory of the perishable good. Goods of age 0, which are either harvested or ordered on each day of the planning horizon, arrive at the major terminal. Some of these goods are immediately shipped to the different DCs. All the quantity on storage in the terminal and the DCs ages of one in every period. Goods of age 0 can be delivered to the final customer on the same day still, which means that first and second level delivery takes less than a full period. Of course, the amount delivered to all customers assigned to a DC can not be larger than the quantity stored there plus the shipments from the major terminal. While the major terminal has unlimited storage space, the DCs have a maximum storage quantity that cannot be succeeded. The wasted amount at each vertex on each day is equal to the quantity of goods with an age of m-1 at the end of the previous period. Prader additionally includes an optional constraint which guarantees that the stored quantity at each storage location at the end of the planning horizon is larger or equal the starting inventory, which might be required to guarantee demand satisfaction in the next planning period. The inventory variable is non-negative which means that stockouts are not allowed.

The next set of constraints in Prader generate the delivery patterns, which in the original work are modeled as following: For each final customer, a set of acceptable delivery days out of the periods of the planning horizon is given. Some consumers want to be delivered once per week, others twice, with a fixed demand quantity that is given. The model picks the requested number of delivery days out of the ones that are acceptable for the customer in a waste minimizing way.

The routing constraints make sure that no customer is delivered directly from the terminal, thereby enforcing consolidation. They also make sure that the capacities of storage locations and vehicles are not exceeded on both levels. Each customer and DC is not visited more than once per delivery period and the number of tours from the terminal and each DC is limited by the number of vehicles available in the respective location.

There is also a constraint that defines a maximum order quantity for each period to model a case in which alternative suppliers also have limited capacities.

5 Solution Method

The mathematical formulation in Prader (2022) only allows for an optimal solution of small instances of up to 12 customers. For this reason a matheuristic is introduced to solve bigger problems. Prader finds a complete solution to the problem by optimally solving the first level with a reduced version of the presented mathematical model, including delivery patterns for final customers but without considering second level routing decisions. Based on the found solution, in the next step the second level is solved as a set of Capacitated VRPs (CVRPs), one for each DC in each period. However, Prader proved that the delivery patterns found like this are optimal for the first level, not, however, for the second one. A general improvement is possible by changing delivery patterns in a way that increases the total costs on the first echelon while simultaneously decreasing the second level costs. This last step is implemented as a feedback loop that takes customers out of their current routes and includes them in the route of their assigned DC on a different allowed delivery day. The method by Prader, except for the feedback loop that is newly added, is based on the one presented by Rohmer et al. (2019). Of course, there is some additional complexity compared to Rohmer because multiple DCs are used instead of a single one and some additional considerations are included.

Generally worth mentioning is that the method by Prader includes two different ALNS algorithms. The first one is used to improve the routes for the CVRPs on the second level after generating them by a cheapest insertion algorithm. Prader call this ALNS the ALNS to generate second level routes, short ALNS-2LR. The second ALNS is part of the feedback loop and alters the days on which final consumers are delivered. Prader refers to it as ALNS to change delivery patterns, or ALNS-DP.

All these steps will be briefly summarized in the following chapter followed by the methodological novelties, mainly consisting of a detailed description of the new ALNS to change customer assignments (ALNS-CCA).

5.1 Summary of the original solution method

The solution method by Prader consists of four main steps that will be described briefly in this section. The following summary is adapted from Prader (2022).

- **Step 1:** A reduced version of the mathematical formulation is solved optimally to find:
  - ⋆ first level routes
  - ⋆ delivery quantities and, thus, storage quantities at each DC in each period
  - ⋆ quantity ordered at the major terminal in each period

  - ⋆ customer delivery patterns based on all information except for second level routing decisions

  The reduced formulation is optimally solvable even for large instances of 400 and more customers. In Prader as well as in the present work the first level is modeled as a TSP which means that a single vehicle delivers the DCs to reduce the complexity of the problem. However, small changes of the model would be sufficient to extend this to a case with multiple vehicles.

- **Step 2:** A cheapest insertion algorithm is used to find starting second level routes for each DC on each day. Which customers are delivered by each DC in each period is fixed after the determination of delivery patterns in Step 1 since customer assignments to DCs are given at this point. This leads to a total of |DC| x |T|-1 independent CVRPs to solve.

- **Step 3:** The starting second level routes are improved by the ALNS-2LR, that is partially based on the ALNS for 2E VRPs from Hemmelmayr et al. (2012) and the original ALNS formulation of Pisinger and Ropke (2006).

- **Step 4:** The feedback loop, including the ALNS-DP, is used to find better delivery patterns and, thereby, improving the complete solution.

5.2 Summary of the three ALNS algorithms

Step 1 and 2 in the previously presented overview are sufficient to generate a complete, but probably sub-optimal, solution to the formulated 2E-IRP. Stopping the solution method at this point would be equal to just solving the two levels of the supply chain independently. However, there are two main connection points between the two levels: the customer assignment to DCs and the delivery patterns, so the days on which final customers are delivered. In a case where these two elements are fixed, the first level is optimally solvable through the reduced mathematical formulation. The second level, on the other hand, reduces to the mentioned |DC| x |T|-1 independent CVRPs, which might be optimally solvable if the number of customers assigned to a DC is not too high. Otherwise a solution by generating a starting solution for the routing and an iterative improvement of the routes by the ALNS-2LR might at least generate a heuristic solution that is close to the optimal.

The three presented ALNS algorithms work together in the following way:

- **The ALNS-2LR** might be necessary to generate a complete solution to the problem for larger instances while small ones are optimally solvable if customer assignments and delivery patterns are fixed and given. As soon as a starting solution for the full problem is found, the other two ALNS algorithms try to find additional improvements.

- **The ALNS-DP** tries to improve the complete solution by changing the delivery periods of final customers.

- **The ALNS-CCA** aims to decrease second level routing costs by changing the assignment of customers to the given DCs.

To sum up, in the adapted solution method we will try to improve the complete starting solution found after Steps 1 and 2 by applying the ALNS-DP and the ALNS-CCA. We also search for a way to combine these two improvement strategies efficiently.

5.2.1 The ALNS-2LR

As already mentioned, for a case with a fixed starting customer assignment as well as fixed delivery patterns, the complete problem might be optimally solvable for instances up to a certain size. Which size that is will generally be examined in the Computational Study, chapter 7. However, as soon as the optimal solution becomes too computationally intensive to be reasonable, the second level is solved heuristically. Here, after the generation of starting routes for each DC in each period, the ALNS-2LR comes into play. It works on a set of CVRPs that are independent from each other. For each of them, a destroy operator first takes customers out of their current route and puts them into the customer pool. In the repair step, the customers from the customer pool are reinserted into different routes based on the applied repair logic. Promising solution found like this are, if possible, additionally improved by a move operator. The ALNS-2LR falls back on a set of different destroy and repair operators which are selected by a roulette wheel procedure. Their chances of being selected depends on their past success in finding improvements.

All these steps are summarized by the following pseudo-algorithm reprinted from Prader (2022), where D represents the set of destroy operators and R the repair operators:

Algorithm 1 ALNS-2LR

$s \leftarrow InitialSolution, InitializeScores(\pi)$
while Stopping Criterion Not Reached **do**
  $N_D \leftarrow ChooseDestroyOperator(D, \pi)$
  $N_R \leftarrow ChooseRepairOperator(R, \pi)$
  $s' \leftarrow DestroyAndRepair(s, N_D, N_R)$
  if $f(s') < (1 + a)f(s*)$ **then**
    $s' \leftarrow ImproveWithMoveOperator(s')$
  end if
  if $f(s') < f(s)$ **then**
    $s \leftarrow s'$
  end if
  if $f(s) < f(s*)$ **then**
    $s* \leftarrow s$
    $UpdateScores(\pi)$
  end if
end while

At the end of a run, the best found solution s* and some other indicators, like the

run-time, are returned.

Summary of destroy and repair steps

The ALNS-2LR, as already mentioned, solves a set of CVRPs. For each of them, starting routes are given after a standard cheapest insertion procedure. The routes for each CVRP are represented by a dictionary with the route number as a key and a list of customers in the order they are visited as a value. The following example shows a CVRP solution where only two vehicles, vehicle 0 and 1, are needed to deliver all assigned customers in the represented period. If we assume that this are the routes of DC 1 in a certain period, the first vehicle leaves the DC, visits customer 219, then 398 and so on and goes back to DC 1 after visiting customer 356. A total of 7 vehicles is available in the example.

{0: [219, 398, 344, 375, 356], 1: [19, 289, 380, 230, 277, 287, 312, 290, 292, 395, 293, 404, 391, 332], 2: [], 3: [], 4: [], 5: [], 6: []}

One of the following destroy logics is now used to pick a given number x of customers, take them out of the solution and put them into a customer pool:

- **Random Removal:** operator from Pisinger and Ropke (2006) that selects x customers randomly.
- **Random Tour Removal:** operator based on Hemmelmayr et al (2012) removing all customers from a randomly chosen tour.
- **Worst Removal:** customers with the highest removal gain are put into the customer pool. The removal gain is the cost saving generated by removing the customer from its current tour, so the sum of the travel costs from its predecessor to the customer and the travel costs to its successor, minus the costs of going from the predecessor directly to the successor.
- **Worst Removal from each Tour:** The same destroy logic as in the previous operator is applied to remove x customers from each tour individually. Nothing is removed from tours with less than x customers in them.
- **Remove Related:** The x closest customers to a random seed customer are removed from the solution. The measure of closeness are the travel costs.

After this removal step, repair operators are used to reinsert the customers from the customer pool into the solution. Some of them insert customers individually and do this only in feasible positions, which means that all capacity restrictions still need to hold. Others insert a set of customers simultaneously, which makes an additional step to make the solution feasible again, necessary, which will be a split operator. The following repair operators are implemented in Prader (2022):

- **Greedy insertion each single customer (with and without perturbation)**: customers from the pool are individually inserted in their cheapest feasible insertion

position. Insertion costs are defined as the cost increase caused by the insertion of the customer in a certain position. A version of this operator including perturbation is also generated in which the insertion costs of each customer are multiplied by a random number between 0.8 and 1.2 to include some randomization.

- **Greedy insertion whole pool (with and without perturbation):** The same procedure as in the previous operator is done not for single customers but for the customer pool as a whole or a set of customers. Again, a version with and without perturbation exists. Here the outcome is often not feasible.
- **Greedy insertion forbidden:** While the insertion logic is the same as for greedy insertion each single customer, here an insertion of a customer into the same tour as before the destroy step is not allowed. If it is required to guarantee feasibility of the solution, the customer is inserted in an empty tour.
- **Regret Insertion:** A regret value is calculated for each customer by summing up the cost difference of inserting it into its cheapest position compared to the following k positions. Customers are then sorted by this regret value and inserted, in decreasing order, resorting to greedy insertion each single customer.

After this step the solution again contains all required customers. If it is infeasible at this point, a split operator makes sure that the tours comply with all capacity constraints again. This is done by generating a giant tour and cutting it whenever adding the next customer would succeed the maximal allowed capacity.

For instance, for the following giant tour with given capacity use for each customer, the following splits would be performed for a case with a maximal capacity of 30:
giant tour: [1,2,3,4,5]
capacity use (same order): [20,10,30,10,10]
split tours: [1,2], [3], [4,5]

After this step, Prader additionally integrates a local search consisting of a classical move operator. This operator starts with the first customer and tries to find a cheaper insertion position for it in the current solution. If there are better positions for it, it is inserted in the cheapest of them, otherwise it stays in its position and the operator moves on to the next customer. This operator proved to be very efficient in finding improvements in Prader (2022).

Relevance of the ALNS-2LR for the present work

As previously mentioned, the ALNS-2LR is used to find a starting solution to the complete problem for bigger instances while it might not be necessary to use it for small instances in the present master thesis.

Since the focus of this study lies on the improvement steps, we refer to Prader (2022) for further details on the ALNS-2LR. The next two operators, however, will be described in more detail. While the ALNS-DP originates in Prader (2022) and will be improved and tested thoroughly in chapter 7, the ALNS-CCA is a new procedure and will be the main focus of the remainder of this work.

5.2.2 The ALNS-DP

At this point of the procedure, a complete solution to the problem was obtained by solving the two echelons mostly independently. For instances up to a certain size this solution might even be optimal if a fixed customer assignment to DCs as well as given delivery patterns are assumed. Otherwise, if the second level is solved heuristically, in most cases the solution can still be accepted as close to optimal. The goal of the feedback loop including the ALNS-DP as well as the ALNS-CCA is to find delivery patterns or customer assignments that lead to a better solution. A change of customer assignments will mainly work on the second level route length, while changing the delivery patterns might reduce the amount of waste occurring in the planning horizon as well as (potentially smaller) reductions in transportation and storage costs.

The ALNS-DP uses the previously found complete solution to the problem as a starting point and applies destroy and repair operators on it. The solution here is again represented by a dictionary, but now it contains another dictionary with the routes of each available vehicle at every DC in each period. This means the problem that the ALNS is working on is considerably bigger and more complex. The destroy operators of the ALNS-DP now take a given number of customers out of all the tours they are currently included in and puts them into the customer pool. A customer that is delivered twice a week like this is taken out of two tours and put into the customer pool. Repair operators include this customer into the routes of two of its acceptable delivery days, which might be the same as previously of different ones.

Operators for the ALNS-DP

As the ALNS-2LR, also the ALNS-DP resorts to a roulette wheel selection to pick a destroy and a repair operator in each iteration. The destroy operators that Prader applies here use a similar logic as the ones from the ALNS-2LR, but on a more complex problem. To be precise, the operators random removal, random tour removal, worst removal and remove related were adapted to the larger problem. Clearing whole tours, on the other hand, does not seem reasonable if some customers are delivered several times in the planning horizon because it would destroy the solution to a very high extend.

Concerning the repair operators, the logic is again similar to the ones from the previously presented ALNS. However, inserting sets of customers together is not applicable here since they might not require the same delivery days. Also, especially for customers that are delivered more than once per week, the greedy insertion forbidden operator does not work. This is especially true since a customer might request delivery on two days out of a set of three, which makes it impossible to avoid all previously selected delivery days.

The ALNS-DP only produces feasible solutions, which makes the split operator irrelevant. However, the local search performed by the move operator is again used as in the ALNS-2LR.

Summary of the feedback loop

In the work of Prader (2022), the ALNS-DP is one component of a feedback loop that aims at finding delivery patterns that save waste and reduce environmental costs. This feedback loop consist of different phases, which are summarized in the following pseudo-algorithm reprinted from „Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective“, by Prader, R., 2022., p.30.

Algorithm 2 Feedback Loop

$s2 \leftarrow StartingSolution2ndLevel$
$s1 \leftarrow StartingSolution1stLevel$
$p \leftarrow CurrentDeliveryPatterns$
while Stopping Criterion Not Reached **do**
  $(p', s2') \leftarrow ChangeDeliveryPatterns(p, s2)$
  if $f(s2') < f(s2)$ **then**
    $s1' \leftarrow Solve1stLevel(p')$
    if $f(s2') + f(s1') < (1 + a) * (f(s1) + f(s2))$ **then**
      $s2' \leftarrow ALNS - 2LR(s2')$
    end if
    if $f(s2') + f(s1') < f(s2) + f(s1)$ **then**
      $s2* \leftarrow s2'$
      $s1* \leftarrow s1'$
    end if
    if $f(s2') + f(s1') < (1 + b) * (f(s2) + f(s1))$ **then**
      $s2 \leftarrow s2'$
      $s1 \leftarrow s1'$
    end if
  end if
end while

At the beginning, the complete solution consisting of the solution of the first level as well as the solution of the second level, is given for the current delivery patterns. In each iteration, first the ALNS-DP (in the algorithm labeled as *ChangeDeliveryPatterns*) is called to change the current delivery patterns and find the associated second level costs. If an improvement is found on the second level, the first level is solved with the new delivery patterns as an input. If a global improvement was found (or a degradation of at most a in the pseudo-algorithm), the ALNS-2LR is used to eventually improve the second level routes further. If a global improvement is found after this step, the new delivery pattern is saved as current best. As long as the new solution is only slightly worse than the previous one (of a value b), it is used as a new incumbent for the next iteration. Otherwise it is discarded and the old incumbent solution is destroyed and repaired again.

In the computational study of Prader it was shown that this feedback loop, because of the complexity of the underlying problem, is computationally expensive. For this reason reducing its runtime will be part of the experiments in chapter 7 of the present work.

5.2.3 The ALNS-CCA

The novel ALNS-CCA is applied on a complete solution based on a starting customer assignment and fixed delivery patterns that were either given (in a case were customers decide about when they want to be delivered) or determined optimally for the first level by solving the reduced mathematical model from Prader (2022).

There is one main difference between the ALNS-DP and the ALNS-CCA that is important to mention at this point: For the ALNS-DP, where the customer assignment is fixed and the starting delivery pattern is the optimal one for the first level, the direction in which the solution changes is known. Since the starting solution is optimal for the first level, every change in the delivery pattern will only increase the costs here. For this reason an improvement can only be found if the second level is improved sufficiently to outweigh the cost increase on the first one. So a total improvement will always have higher first level and lower second level costs.

For the ALNS-CCA, the starting assignment of customers to DCs will be generated by one of three developed algorithms. If any change of the initial clustering will improve or worsen the first or the second level solution is not clear from the beginning. Therefore each alternation of the customer assignment requires a reevaluation of both levels.

In general, the novel ALNS algorithm is constructed as following: First of all, a complete solution is obtained for the initial clustering. The clustering is represented by a dictionary with the DCs as keys and a list of the currently assigned customers as a value. A roulette wheel selection is used to select a destroy and a repair operator and they are applied on the solution as in the previous two algorithms. The first level is then solved for the modified assignment, since in most cases this is less computationally expensive than solving the second level. If the first level solution is better or only slightly worse, the second level is solved again, too, otherwise a new iteration starts using the previous assignment. If, after solving the second level again, a total improvement is found, the new customer assignment is kept as a new incumbent for the next steps.

Please note that the first and the second level are solved with the same method as for finding the complete starting solution (which is similar to version 1 of the solution method in Rohmer et al. (2019)):

- **The 1st Echelon solution** is found by solving the reduced mathematical formulation optimally in most cases. For large instances, the solution, obtained by the gurobipy solver, can be stopped at a certain percentage from the optimal if necessary.
- **The 2nd Echelon solution,** consisting of several CVRPs generating the routes of each DC in each period, can be solved optimally for instances up to a certain size. For bigger instances starting routes are generated by a cheapest insertion algorithm and improved by the ALNS-2LR.

The following pseudoalgorithm summarizes the steps of the ALNS-CCA:

Algorithm 3 ALNS-CCA

$CA \leftarrow GenerateStartingAssignment$
$s1 \leftarrow SolveFirstLevel(CA)$
$s2 \leftarrow SolveSecondLevel(CA)$
$\pi \leftarrow InitializeScores$
while Stopping Criterion Not Reached **do**
  $N_D \leftarrow ChooseDestroyOperator(D, \pi)$
  $N_R \leftarrow ChooseRepairOperator(R, \pi)$
  $CA' \leftarrow DestroyAndRepair(CA, N_D, N_R)$
  $s1' \leftarrow SolveFirstLevel(CA')$
  if $f(s1`) < (1 + a)f(s1)$ **then**
    $s2' \leftarrow SolveSecondLevel(CA')$
    if $f(s1') + f(s2') < (1 + b)(f(s1) + f(s2))$ **then**
      $s1 \leftarrow s1'$
      $s2 \leftarrow s2'$
      $CA \leftarrow CA'$
    end if
    if $f(s1') + f(s2') < (f(s1*) + f(s2*))$ **then**
      $s1* \leftarrow s1$
      $s2* \leftarrow s2$
      $CA* \leftarrow CA$
      $UpdateScores(\pi)$
    end if
  end if
end while

Since we do not want to fix the number of customers the algorithm is allowed to assign to a DC, we do not include capacity considerations into the repair operators. However, we make sure that the capacity is not exceeded by interrupting iterations that lead to a break of the capacity constraints and start a new iteration based on the last accepted incumbent.

The general parameters are chosen similarly to Prader (2022). At the beginning, all operators have the same chance to be chosen and each has a score of one. As soon as an operator finds an improvement that is accepted, this score is increased by one which makes it more probable that such a successful operator is chosen again. The value of a in the algorithm is set to -0.1 while b is set to 0 in most runs, which means that we only continue with a new solution if an improvement was found.

All operators that remove customers separately in each run delete a random number of customers between 0 and 1/10 of the total number of customers. Operators that remove all customers from a DC do that for exactly one DC because otherwise the solution is changed to much in one iteration.

Operators to change the customer assignment

The new ALNS developed in this work includes different measures of closeness in the greedy insertion as well as the worst removal operators. Finding out which of them find most improvements will give valueable information about what are the characteristics of an efficient clustering for different problem settings.

The following measures of closeness are tested in the present work:

- **Simple removal gain:** of a customer is the distance or travel time from the customer to the DC it is currently assigned to.
- **Complex removal gain:** is the sum of the travel times from a customer to all other customers assigned to its current DC.
- **Simple insertion costs:** of a customer to a DC are defined as the travel time or distance between them.
- **Complex insertion costs:** of a customer to a DC are again the sum of travel times of the customer to all other customers that are currently assigned to a DC.

The destroy operators used in the ALNS-CCA are the following:

- **Random customer removal:** a given number x of customers is randomly selected for removal from their current DC and put into the customer pool.
- **Random DC removal:** one DCs is picked randomly and all its currently assigned customers are removed and put into the customer pool.
- **Worst customer removal (simple or complex):** the x customers with the highest removal gain (either simple or complex) are removed from their current assignment.
- **Worst removal from each DC (simple or complex):** the x customers with the highest removal gain are removed from each DC that has more than x customers assigned currently.
- **Remove related customer:** a random customer and its x closest customers are removed from the solution and put into the customer pool.

Customers are reinserted into the solution by the following repair operators:

- **Greedy insertion each single customer (simple and complex, with and without perturbation):** the customer pool is randomly shuffled and the customers are inserted one by one into their position with the lowest insertion (simple or complex) cost. All insertion costs are also perturbed by multiplying them with a random number between 0.8 and 1.2 in the perturbed versions of the operator.
- **Greedy insertion each single customer forbidden (simple and complex):** Same insertion logic as previously but an assignment of a customer to the same DC as before the destroy operator is not allowed.

- **Regret insertion (simple):** The customer pool is sorted by non-increasing regret value, which for this operator is calculated as the difference between the lowest and the second lowest simple insertion costs. Greedy insertion each single customer simple is now run on the sorted customer pool. This increases the chance of customers with a high regret value to be assigned to their closest DC.

Value added by the ALNS-CCA

The goal of the ALNS-CCA is to find a cheaper and more environmental friendly solution to the 2E-IRP from Prader (2022) by improving the customer assignment to DCs. Implementing this as an improvement algorithm that is independent of the complete solution of the problem has the additional advantage, that it can be used more flexibly and in the most suitable way for the current instance and case.

It also helps to find a problem specific clustering of customers to the DCs while all capacity restrictions are taken into consideration. Having such an independent clustering algorithm is especially helpful if one DC needs to be closed or is not reachable in one planning horizon, which might be the case after natural disasters. It helps to quickly and efficiently relocate its customers to the remaining DCs.

Also, in a case where the street conditions are subject to change, the optimal customer assignment might change short term. The flexibility of the ALNS-CCA helps to react fast to such conditions, as the computational study will show.

6 Instance Generation and Model Adaptations

Two echelon supply chains are common in different contexts. One is, as in the original work, urban transportation, where large vehicles with a high speed but also high environmental costs, deliver intermediate depots. Since such large vehicles are not very agile in small urban alleys, the mean of transport in such a setting is often changed, in our case at the DCs. From there, smaller vehicles with lower capacity, lower (or no) environmental costs, but eventually also less protection from sun, heath or other externalities, deliver the final customer. For this reason, even though the food items do not age in days on this last mile delivery step, we assume that the quality of the good decreases in this step. The level of decay is dependent on the transportation duration.

For most of the instance sets used in the computational study, the following settings are considered, in all eight possible combinations:

- **small and large size instances**: The small size instances have a total of 50 customers to deliver and an optimal solution of both levels for a case with fixed customer assignment and fixed delivery patterns is still possible. For such a solution the problem is deconstructed and each level is solved as a MILP by the gurobipy solver. The fact that the two connecting factors, the customer assignment and the delivery patterns, are assumed to be fixed, allows for this deconstruction of the problem. Large size instances include 400 final consumers and the second level is solved by a cheapest insertion algorithm in combination with the ALNS-2LR.

- **small and large grid size**: Small grid size instances deal with a delivered area of a size of 10 times 10 measures of distance while in large size instances customers are spread over a 100 times 100 distance measure large square. The grid size impacts the relations of the cost components of the objective function. In a small grid the transportation costs will be relatively lower than storage and waste costs.

- **major terminal location**: The major terminal is positioned either at the corner of the grid or at a random position. This will influence the route lengths on both levels and, thus, the proportion of the routing costs.

Table 6.1 summarizes the versions of the instance sets that will be used for tests in the computational study.

In the table, the instance name consists of the first part, which states the number of customers N, the second part which classifies the grid size G as small (s) or large (l) and the terminal position T that is either at the bottom left of the grid (bl) or random (r). To distinguish the different instance sets an additional letter will be added as following:

Data Set	Properties			
	Customers	DCs	Grid size	Terminal Position
N400-Gs-Tbl	400	6	10x10	bottom left
N400-Gl-Tbl	400	6	100x100	bottom left
N400-Gs-Tr	400	6	10x10	random
N400-Gl-Tr	400	6	100x100	random
N50-Gs-Tbl	50	3	10x10	bottom left
N50-Gl-Tbl	50	3	100x100	bottom left
N50-Gs-Tr	50	3	10x10	random
N50-Gl-Tr	50	3	100x100	random

Table 6.1: 8 Versions tested for most instance sets

- **U**: Urban food delivery
- **LM**: Lower-middle income level countries
- **V**: Delivery of villages
- **CQ**: Changing street quality
- **DZ**: Disaster zone delivery

The following sections give an overview of the properties of the newly generated instances for the computational study. Even though it is impossible to fully exploit all types of 2E supply chains, this shall show how flexibly adaptable the model and the presented solution method are to different problems dealing with food waste reduction and food safety.

For the general tests there are some things that all large instance versions have in common. The deterioration age m is 3 time units and the total number of periods in the planning horizon is 5 days. There is one single vehicle available in the central terminal, which makes the first echelon routing problem a travelling salesman problem and decreases the complexity. However, this could be changed by some simple adaptations of the model as explained in Prader (2022).

The vehicle capacity is 8500 units on the first echelon and 750 units on the second one while the wage is 0.001 monetary units per meter. The storage capacity of the DCs is of 2500 units. The demand of all customers is generated randomly between 1 and 100 units and the supply at the terminal is a random number between 6000 and 12000, based on the experiments in Prader (2022). Holding costs are of 0.005 monetary units per unit in the DCs and 0.007 units more expensive in the major terminal for goods of age 0. These costs increase of 0.001 for every day a good ages. Such age-dependent holding costs were used in Rohmer et al (2019) and Prader (2022) to penalize the delivery of CTD products to the customer and, therefore, reduce the chance that goods are wasted at the final customer.

The penalty costs for goods wasted in the supply chain are 0.01 monetary units per piece and order costs are generated randomly for each day of the week between 0.006 and 0.04 monetary units per piece. As in Prader (2022), the first half of the customers is visited once per week while the second half requests to get their demand twice per week. The combination of days on which customers are available for delivery is generated randomly as three or four days of the week.

6.1 Instance Set 1: Urban food delivery

The first instance used in this work is, as in the original work of Prader (2022), urban food transportation on two stages. On the first level, big combustion engine vehicles with efficient cooling systems are used. Also, we assume that the high speed of those vehicles and the relatively short distances traveled make the deterioration during the first level transportation neglectable. However, the second level is delivered by more environmentally friendly but also much slower vehicles like cargo bikes, in which efficient cooling might not be guaranteed. We assume that long transportation distances here have an impact on the freshness of the goods and, therefore, on the customer satisfaction and the waste level at the final customer. While Prader (2022) assumed that the low transportation speed makes it reasonable to fix the assignment of customers to the DCs, we will not fall back on this assumption. One main reason is that it might not always be the distance from the DC that is relevant to decrease the route length but the distance between customers. To illustrate this let us assume a case where two customers are very close to a DC but on opposite sites. Another customer might be further away from the DC but since the travel distance from one of the other customers is short, combining them might still make sense. Because of this reasoning we will also test a second distance measure, the complex insertion costs and removal gain. It is not based on the distance of the customer to its current DC but on the distance from the customer to the other customers assigned to its current DC. In chapter 7 we will test if destroy and repair operators working with this distance measure are more or less successful than the ones falling back on the old cost calculation.

The instances considering urban transportation will mainly differ in one thing from the ones in Prader (2022): we will not consider the assignment of customers to DCs as fixed but change it with the ALNS-CCA. Also, we will use the eight versions of the instance mentioned in table 6.1, for our tests.

6.2 Instance Set 2: Lower-middle income level countries

Decreasing the length of the second level routes becomes even more relevant for the second set of test instances, that deals with a similar transportation system in lower-middle income level countries like Indonesia, Lao, Nepal or Vietnam. Here it is common that some important streets, that in our case connect the central depot to the DCs, have relatively high standards, while streets within and to smaller villages, and therefore to the final customers, can be of poor quality. Oftentimes the only means of transport here

are mules or human carriers, fully loaded scooters or eventually off-road vehicles (so GHG emissions might be relevant on the second level again). This makes the transportation on the second level slow, which might lead to an additional environmental damage since many goods deteriorate on this step. This is even enhanced by an eventual impracticality of cooling perishable goods properly in most of the here used vehicles. Depending on packaging and the mean of transport, factors like air pollution or the dust stirred up from dirt roads might additionally contaminate food items.

For this instance set, the transportation costs for each edge on the second level are assumed to be an estimator including such factors. This means that the deterioration during transportation is integrated in the costs of each arc instead of adding them, as an additional factor, to the objective function. Even though we are aware of the fact, that this is a strong simplification, no general term is known to us that summarizes the mentioned deterioration, depending on different external factors, sufficiently.

Customer locations will, as for instance set 1, be generated randomly in the grid. The main difference between instance set 1 and 2 will be the costs, that now additionally represent the delivery speed as well as the dimension of food waste on the second level. To sum up, for instance set 2, we generate a new random instance but with second level transportation costs that are higher because they include costs of deterioration and degradation of food safety.

6.3 Instance Set 3: Delivery of villages

The third set of instances is generally similar to set 2 but now the customer locations will not be generated completely randomly but in small clusters, representing small villages spread over the grid. The distances between such villages as well as their size will be varied. One goal here is to find out how this influences the optimal customer assignment to DCs and the second level routing costs.

6.4 Instance Set 4: Changing street quality

The fourth instance set will deal with the fact that the street quality might change in different periods in some countries. In Nepal, for instance, landslides and floods as well as frequent earthquakes might be the reason of changing street conditions. In other countries some streets might be flooded, or simply very hard to cross, from time to time in the rainy season. Countries like Lao, Cambodia, Myanmar, Vietnam and Thailand also deal with something that is called the *smoke season*, where farmers burn sugarcane fields to fertilize them, or forests to ease the quest for expensive mushrooms. Some areas in this time have a dangerous level of air pollution which might contaminate the food as well as put the safety of drivers at risk.

To simplify the problem, in this set of test instances we will assume that the street quality is given in the form of travel cost estimates for every arc that will be multiplied with the arc's length. In this setting, the shortest tour might become the least efficient if

it crosses many though street parts, which is an additional argument for not fixing the customer assignment to DCs.

To be precise, for this instance set we will find a close to optimal customer assignment for given costs of each arc. Afterwards, we assume that the costs of some arcs change, for instance because of heavy rainfall in the rainy season, that affects most of the tropics and therefore around one third of the world population. We will calculate the new costs of the old solution and see how much they can be improved by changing the customer assignment based on the new costs with the ALNS-CCA.

6.5 Instance Set 5: Disaster zone delivery

The last instance set is an extreme case of set 4, simulating a case where some streets have become impenetrable because of floods, earthquakes, fires or other natural disasters. Street conditions, or major damage, might cut off one or several DCs from the transportation system. This can lead to a scenario where some DCs cannot deliver any customers anymore. Consequentially, all their customers have to be supplied by the remaining DCs.

In the mathematical formulation this only changes that the affected DCs are taken out of the set of available DCs. However, as the computational study will show, this increases the complexity considerably and makes an optimal solution impossible. The novel ALNS-CCA allows us to solve the problem as previously in such a case and simply reassign the customers of the unavailable DC.

Of course, the new problem generated in instance set 5 is only feasible if the storage and delivery capacities of the remaining DCs are sufficient to fulfil the supply of all customers.

7 Computational Study

As already mentioned, the main focus of the present work is on the improvement algorithms, so the ALNS-DP and the ALNS-CCA. The ALNS-2LR was already tested extensively in Prader (2022) and will be used to generate a complete solution to a problem when delivery patterns and customer assignments to DCs are fixed. For this reason, this algorithm is not changed compared to Prader.

The aims of the present computational study are the following:

- 5 sets of test instances as described in chapter 6 are generated in versions with varying customer number, grid size and positioning of the major terminal.
- The success of the ALNS-CCA in finding efficient customer assignments to DCs is assessed. This will be measured by its ability to find reductions of the costs of food waste, food insecurity and environmental damage caused by transportation and storage compared to three differently generated starting assignments.
- As soon as a customer assignment is accepted, additional improvements are persued by the feedback loop by Prader including the ALNS-DP. In this process we also aim at adapting the ALNS-DP in order to make it less computationally expensive.
- Another objective of the present work is to find a way to effectively combine the ALNS-CCA and the ALNS-DP. The ALNS-DP works on a solution with a fixed customer assignment while the ALNS-CCA deals with a solution with fixed customer delivery patterns. Therefore, finding a good solution where both are as good as possible might be hard to balance.

Prader proved that their solution method, consisting of the generation of a complete solution and afterwards improving it with the feedback loop, was able to find the optimal solution of a problem with fixed customer assignments for small instances of up to 12 customers. Their MILP is not optimally solvable in a reasonable time frame for larger instances anymore.

Finding an optimal solution for a case where customer assignments are not fixed anymore is an even more difficult problem and already for a very small instance of 10 customers and 2 DCs the complete MILP takes gurobipy over 3000 seconds to solve to 2% from optimal. Finding the optimal is not possible anymore in a reasonable time frame by solving the MILP. The same instance, but with fixed customer assignments, takes under four seconds to solve.

There are two main implications from the fact that an optimal solution with changing customer assignments is so much more computationally expensive:

First of all, it makes a comparison to the optimal found from the solution of the MILP difficult. However, as we will see later, the deconstruction of the problem into two levels that are connected by delivery patterns and customer assignment allows for an optimal solution of the complete problem if the two connecting factors are fixed, at least for instances up to a certain size. In the following sections, we will compare such an optimal solution with 50 customers to the heuristic one to evaluate the ALNS-CCA.

Secondly, it shows that finding improvements of the total solution by changing customer assignments heuristically is necessary because solving the problem as a whole is so hard. Also, in previous works from literature, like Rohmer et al. (2019), the solution that we use as a starting solution is already accepted and no further improvement is striven for. This means, that every improvement found compared to such a starting solution already adds something to previous study results.

The complete computational study was run on a single machine with an AMD Ryzen 7 5700U processor at 1801 MHz. All the coding was done in Python 3.8.12 on the scientific environment Spyder IDE. The MILP was solved by the Python gurobi interface gurobipy.

In general, the computational study is split into three main parts. The first one mainly deals with the new ALNS-CCA and does a general parameter setting for it. In those first runs we will assume that customer delivery patterns are fixed and given and we will only try to optimize the customer assignment to DCs. It is important to mention, that the waste quantity happening like this during storage (and because of the tracked product age) does not change, but mainly the transportation costs, which now also include the deterioration during transportation. In addition to that, this first part will also evaluate the success of the new destroy and repair operators and the two different used measures of distance. All these tests are performed on instance set 1, which deals with the urban transportation case from Prader (2022).

The second part of the computational study consists of running several tests on instance sets 2 to 5 to see how successful the ALNS-CCA is on their individual conditions and settings.

Finally, the assumption that the delivery patterns are fixed is omitted and we instead investigate a case where the initial delivery patterns are customer preferences but can eventually be changed because of the environmental awareness of consumers. Changing the delivery days of individual customers can, as Prader proved, reduce the environmental impact of the supply chain and the waste quantity. The focus in this final part of the computational study is on trying to find an efficient way to combine the improvement algorithms to find a balanced solution with good customer assignments and good delivery patterns. This is done by, first, fixing both, and then improving the customer assignment as much as possible or reasonable. The solution found like this is than additionally improved by the feedback loop from Prader (2022). In this last part we will also explain how we changed the feedback loop compared to Prader to make it less computationally expensive.

7.1 Instance Set 1 and general parameter setting for the ALNS-CCA

The parameter setting for the ALNS-CCA will be done based on the following four basic steps. Please note that in all of the current section we assume that delivery patterns are given by the customers. In our instances they will either be generated randomly or the patterns that are optimal for the first level found by the reduced mathematical model are used.

- **Step 1:** Find a good way to get a starting solution for the customer assignment, i.e. a good customer allocation that leads to the lowest costs possible. In the computational study different starting solutions will be tested as described in section 7.1.1.
- **Step 2:** Find out if it is necessary to solve the second level with 1000 iterations of the ALNS-2LR (as done by Prader) or if less is enough. Test the algorithm's performance at finding improvements if stopped after a lower number of iterations.
- **Step 3:** Analyze the impact of being able to change customer assignments, measured as cost improvements found by making the assignment flexible. The performance of the ALNS-CCA will be tested for several different stopping criteria: a given number of iterations and 5, 10 or 20 iterations without improvements found.
- **Step 4:** Test the success of the new destroy and repair operators and their combination to find out if the algorithm should combine some of them more frequently.

7.1.1 Initial clustering

In general we deal with a case where the DCs are given as centroids and we need to allocate an unknown number k of customers to them. Not knowing which and how many customers shall be assigned to each DC and which would be an efficient distance measure for the complex underlying problem makes this problem hard to solve. For this reason we start from three different initial assignments and use the ALNS-CCA to improve them. We will experiment with the following three starting customer allocations:

Random customer assignment

The first option is to just assign customers to DCs randomly, we will use a case where each DC supplies roughly the same number of customers in the starting assignment.

Closest customer assignment

Option two is the method used by Prader, which is a simple algorithm that starts with assigning a sorted list of customers to each DC. The customers are ordered by increasing distance to the DC and, starting with DC 1, the first $n/dc + 3$ customers on their lists that are not currently assigned to any other DC are assigned to the present DC. Here,

n represents the total number of customers and dc is the number of DCs. Adding the integer 3 to this number was necessary in Prader to guarantee that all customers are assigned in the end. This algorithm leads to a solution in which all DCs have exactly n/dc + 3 customers assigned except for the last one, which will usually get fewer customers.

Regret customer assignment

The last initial clustering is generated by a regret based algorithm. For each customer, the cost difference of inserting it into its second most distant DC instead of the closest one is calculated and customers are sorted based on their non-increasing regret value. Starting from the first customer on this sorted list, customers are now assigned to their closest DC that still has available capacity. Again, we assume that a new customers can be assigned to a DC until it has a total of maximum n/dc + 3 customers to deliver.

7.1.2 Stopping criterion for the first level

Since for the used instances the first level including all decisions, except the second level routing, is optimally solvable in a reasonable time, it is solved up to this point. For bigger instances gurobipy allows to stop the solution at a percentage from the optimum, which can help to speed up the algorithm further.

7.1.3 Stopping criterion for the second level

Solving the second level to get a complete solution heuristically with 1000 iterations of the ALNS-2LR in Prader took relatively long. For instance, solving a single CVRP with 100 customers like this took over 100 seconds. Since this step has to be undertaken in each iteration of the novel ALNS-CCA, reducing the runtime here is essential. In the following section we will test if solving a lower number of iterations is sufficient to find improvements in the customer assignments or if only using the starting solution found by the cheapest insertion algorithm is sufficient.

7.1.4 General Parameter setting on Instance Set 1

The parameter setting for the ALNS-CCA is done on instance set 1. Afterwards, the here determined parameters will be used in all further tests with the four novel instance sets.

Stopping criterion for level 2

A first major step is determining whether solving the second level with a smaller number of iterations than 1000 is enough to find good improvements. We first experiment with 10 iterations of the ALNS-2LR to see whether that is enough. The starting customer assignment here is generated randomly.

Table 7.1 shows the improvement found with a low number of iterations of the ALNS-2LR, here 10 iterations, compared to a high number to see if it is sufficient for the purpose of the ALNS-CCA. To test this, after each run of the ALNS-CCA with 10 iterations of

Data Set	1st level		2nd level			
	$c_{starting}$	c_{final}	$c_{starting}$	c_{final}	$c_{starting}real$	$c_{final}real$
N400-Gs-Tbl-U	514.9	515.2	1433.8	638.5	1327.7	618.7
N400-Gs-Tr-U	339.1	341.1	1420.4	577.3	1328.0	559.0
N400-Gl-Tr-U	740.3	794.5	13 987.9	5887.4	13 390.6	5723.5
N400-Gl-Tbl-U	1146.2	1184.0	13 698.5	6146.0	13 038.0	5923.7
N50-Gs-Tbl-U	1644.9	1642.2	316.4	205.5	308.1	205.4
N50-Gs-Tr-U	1622.9	1622.2	317.7	193.9	315.2	193.9
N50-Gl-Tr-U	1837.5	1837.5	3093.5	2187.4	3069.2	2186.5
N50-Gl-Tbl-U	1943.0	1943.0	2841.2	2022.5	2804.9	2022.0

Table 7.1: Improvement on both levels found by 50 iterations of the ALNS-CCA. The performance of the algorithm with 10 iterations of the ALNS-2LR to generate the second level solution is tested by comparing the improvement found like this to the "real" solution. "Real" here means the solution with 1000 iterations of the ALNS-2LR for large instances and the optimal solution with gurobipy stopped at five percent from the optimal for the small instances.

the ALNS-2LR, the second level for the starting assignment and the final assignment is recalculated with a higher number of iterations of the ALNS-2LR, in this case 1000, to see if what was found is really as good as an improvement as suggested by the algorithm. The first two columns of the table are the optimally calculated first level costs before (*c_starting*) and after (*c_final*) the ALNS-CCA. Columns three and four are the second level routing costs before (*c_starting*) and after (*c_final*) the ALNS-CCA. The second level is here solved by 10 iterations of the ALNS-2LR. Finally, the last two columns in table 7.1 show the second level costs with 1000 iterations of the ALNS-2LR before and after the change of the customer assignment. In the table we refer to them as *c_starting real* and *c_final real* because these costs are likely to be closer to the real optimal.

The results of these first runs prove clearly, that even if the calculation of the complete solution in each iteration is stopped at an early number of iterations for the ALNS-2LR, we still find very good improvements of the total solution. For example for instance N400-Gs-Tbl-U the first level costs almost did not change at all by altering the customer assignment. They even increased of 0.3 from 514.9 to 515.2. However, on the second level the costs appear to have decreased from 1433.8 to 638.5, which is a total of 795.3 monitary units, if the cheapest insertion solution is improved with only 10 iterations of the ALNS-2LR. Solving it to 1000 iterations shows, that the real improvement is from 1327.7 to 618.7, so of 709.0. In fact, in all runs, the improvement found according to what is referred to as real costs in the table is similarly good as suggested by the algorithm that stops the second level solution earlier.

For the first four instances in the table, which are the large instance versions, the complete run of 50 iterations of the ALNS-CCA took between 93 and 121.5 seconds. On the other hand, the calculation of the second level solution with 1000 iterations of the

ALNS-2LR for the starting and the final assignment alone took between 159 and 197 seconds. This shows, that increasing the number of iterations for the second level to 1000 would increase the runtime of the novel ALNS considerably since this calculation would need to be done in every iteration.

For generating the small instance with 50 customers we did not change the capacity restrictions or any other details of the data set except for reducing the customer number from 400 to 50 and the number of DCs from 6 to 3. This means that capacity is close to unlimited. Here, the second level solution with 10 iterations of the ALNS-2LR is compared to the optimal solution of gurobipy for each DC in each period, however, stopped at five percent from optimal. Again, the comparison shows that the simplified solution is representative for the optimal one. One thing that is noticeable here is that the final costs found with 10 iterations of the ALNS-2LR are the same or extremely close to the real final costs, which suggests that for such a small instance the 10 iterations of the ALNS-2LR might already be enough to find or approximate the optimal.

The runtime here for the complete ALNS-CCA was of 18 to 32 seconds, while just solving the second level of the starting and the final assignment to five percent from the optimal with gurobipy took between 87 to 135 seconds. Especially this comparison to the optimal solution shows that stopping the second level route generation at an early time is a simplification that does not have a strong negative impact on the solution quality but that decreases the runtime considerably. Therefore, this will be kept for the remainder of this work.

The fact that 10 iterations of the ALNS-2LR seem sufficient might lead to the expectation that using just the cheapest insertion solution for second level routing, so the starting solution, could decrease the runtime even further. However, as proved by Prader, cheapest insertion is a very simple algorithm with strongly varying solution quality depending on the instance. For this reason, we decided to keep some improvement by the ALNS-2LR in the final version of the ALNS-CCA.

So far, the starting customer assignment was constructed completely randomly, which leads to fast and easy improvements of the solution by improving the assignment. The next step for the urban transportation instance is to test whether the ALNS-CCA is similarly successful with a stronger starting solution, as the algorithm used by Prader which was described in 7.1.1 as closest customer assignment algorithm. The results are displayed in the same fashion as for the random starting assignment in table 7.2.

For the first four instance versions, the large ones, the number of improvements found in 50 iterations was between 8 and 13. Running the ALNS-CCA took between 107 and 141 seconds, with an average of 125 seconds. On the other hand, solving the second level with 1000 iterations of the ALNS-2LR just for the starting and the final assignment took between 180 and 212 seconds, on average 195.

For the four small instance versions the found number of improvements also ranged from 8 to 13 while the runtime was between 19 and 22 seconds with an average of 21. Solving the second level with the starting and the final customer assignment to optimality with the gurobipy solver took around four times longer, between 82 and 89 seconds with an average of 87.

Data Set	1st level		2nd level			
	$c_{starting}$	c_{final}	$c_{starting}$	c_{final}	$c_{starting}real$	$c_{final}real$
N400-Gs-Tbl-U	513.4	515.6	894.3	606.5	841.6	571.6
N400-Gs-Tr-U	337.9	341.0	869.4	583.9	831.9	555.3
N400-Gl-Tr-U	747.8	795.2	8766.6	5755.1	8335.6	5606.8
N400-Gl-Tbl-U	1156.3	1182.6	9002.3	6010.4	8516.6	5712.1
N50-Gs-Tbl-U	1646.8	1642.6	241.2	203.1	239.4	203.1
N50-Gs-Tr-U	1629.5	1622.2	241.2	204.7	239.4	204.5
N50-Gl-Tr-U	1814.0	1814.0	2413.2	2041.8	2393.9	2041.8
N50-Gl-Tbl-U	1943.0	1943.0	2408.8	2006.9	2393.9	2002.7

Table 7.2: Improvement on both levels found by 50 iterations of the ALNS-CCA, starting from a distance-based customer assignment. The performance of the algorithm with 10 iterations of the ALNS-2LR is compared to the "real" solution.

As expected, the starting costs on the second level are considerably lower than with the previous starting solution, that was generated randomly. However, the ALNS-CCA was still able to find considerable improvements by changing the customer assignment.

Finally, in table 7.3 the regret based starting algorithm is used, again on the same eight instance variants.

Data Set	1st level		2nd level			
	$c_{starting}$	c_{final}	$c_{starting}$	c_{final}	$c_{starting}real$	$c_{final}real$
N400-Gs-Tbl-U	513.4	515.8	901.7	599.5	843.2	577.7
N400-Gs-Tr-U	331.9	340.8	872.0	578.7	834.4	563.2
N400-Gl-Tr-U	747.8	793.9	8900.3	5951.5	8359.3	5582.9
N400-Gl-Tbl-U	1156.3	1182.1	9096.1	6041.2	8446.2	5764.6
N50-Gs-Tbl-U	1646.8	1642.6	241.2	203.2	239.4	203.2
N50-Gs-Tr-U	1629.5	1622.3	241.2	205.5	239.4	205.3
N50-Gl-Tr-U	1814.0	1814.0	2411.9	2036.4	2393.9	2027.7
N50-Gl-Tbl-U	1943.0	1945.5	2408.8	2030.2	2394.0	2025.4

Table 7.3: Improvement on both levels found by 50 iterations of the ALNS-CCA, starting from a regret based customer assignment.

For the instance versions with 400 customers and 6 DCs, between 5 and 14 improvements were found in 50 iterations, with an average of 10. The runtime ranged from 83 to 150 seconds, 117 on average. The calculation of the real costs of the starting and final assignment took between 158 and 164 seconds.

On the other hand, between 6 and 7 improvements were found in 50 iterations on the small instance. The runtime was between 19 and 24 seconds with an average of 20.

Finding the real costs of the starting and the final assignment on the second level took between 72 and 80 seconds, 78 on average.

Not surprisingly, the total improvements found compared to the starting solution are lower compared to the random starting assignment. For the large instances, the previously tested starting assignment from Prader (2022) seems to perform slightly better, since the starting costs on the 2nd level are a a bit higher for this last set of runs. For the small instance they almost did not change, here both starting assignments seem to perform well.

To conclude this section, no matter with which starting assignment the ALNS-CCA was run, it always found considerable improvements. Stopping the ALNS-2LR for the complete solution generation at an earlier point, namely 10 iterations, proved to be successful for all cases.

Stopping criterion ALNS-CCA

The next step in this computational study will be finding a good stopping criterion for the ALNS-CCA. While in the first tests a fixed number of iterations, 50 to be precise, was used, now a given number of iterations without improvement will be used.

Table 7.4 shows the average performance of the ALNS-CCA over 5 runs. A run is ended as soon as no improvement was found in the last x iterations. The values for x that are tested are 5, 10 and 20. For each instance the average runtime as well as the average cost improvement found by the algorithm is displayed for each tested value of x. For this section all runs were started with the regret based customer assignment.

Since it was proven in the previous step, that 10 iterations of the ALNS-2LR are enough to efficiently improve the customer assignment, from this point on we do not compare the result with the the "real" costs but just display the second level result with this number of iterations.

Data Set	5 iterations		10 iterations		20 iterations	
	runtime	avg. imp.	runtime	avg. imp.	runtime	avg. imp.
N400-Gs-Tbl-U	19	270.5	31	282.1	57	298.9
N400-Gs-Tr-U	32	205.0	42	271.8	63	287.1
N400-Gl-Tr-U	26	2261.7	44	2934.1	78	2824.7
N400-Gl-Tbl-U	26	2765.9	39	2935.6	92	2877.0
N50-Gs-Tbl-U	5	38.8	8	38.4	12	41.2
N50-Gs-Tr-U	5	37.4	9	42.4	11	45.3
N50-Gl-Tr-U	6	321.9	7	307.7	12	369.8
N50-Gl-Tbl-U	5	315.8	7	333.4	13	359.7

Table 7.4: Average cost improvement found and runtime of the ALNS-CCA on eight instances over five runs. For each instances, 5, 10 or 20 iterations of the ALNS-CCA without an improvement are used as a stopping criterion.

Not surprisingly, in all cases the solution takes longer the higher the number of iterations without improvement, x, is chosen. For the first two large instances displayed, the found improvement is higher the higher the value of x. However, for instance N400-Gl-Tr-U and N400-Gl-Tbl-U the average improvement found with 10 iterations without improvement is actually higher than the one with 20 iterations without improvement. Even though this might be a coincidence, one thing is generally noticable in the table. Usually, the average improvement found with 10 iterations without improvement is considerably higher than the one found with 5 unsuccessful iterations. However, doubling x from 10 to 20 does not seem to have a strong influence on the improvement anymore in most cases, while the runtime increases noticably.

Since 10 iterations without improvement seem to deliver good improvements in a reasonable amount of time we will keep this as a stopping criterion from this point on.

Graphical representation of the found improvements

The easiest way to analyze the found customer assignment is by representing them graphically. This will, in the following sections, be done by depicting the locations of the major terminal, the DCs and the final customers in a coordinate system. The major terminal here is represented by a red square, the DCs are shown as big green circles and the customers are the small dots. Their colors represent the customer assignment in the sense that all customers assigned to the same DC have the same color. Figure 7.1 shows the random starting customer assignment on the left and a found improved assignment after the application of the ALNS-CCA with the chosen parameter setting.

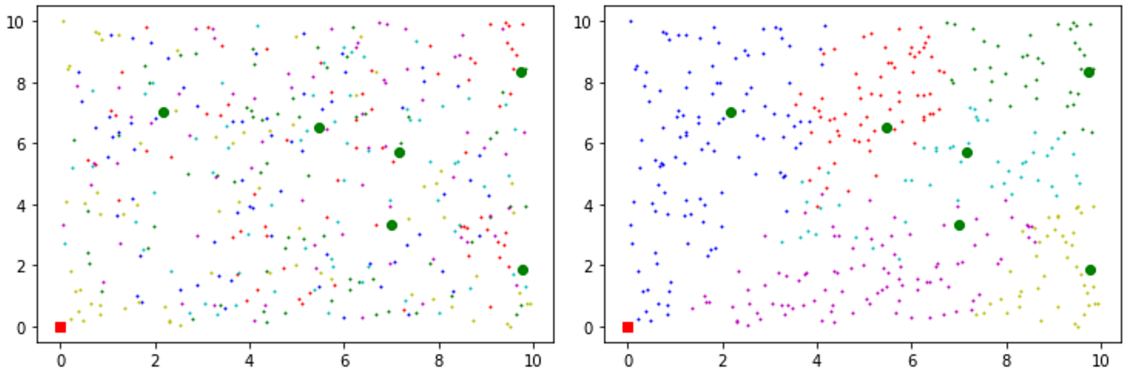

Figure 7.1: Random starting assignment (left) and improved customer assignment after the application of the ALNS-CCA (right).

Success of destroy and repair operators and measures of distance

The following section tests the success, so the ability to find improvements of the solution, of the developed destroy and repair operators as well as their combinations. Table 7.5

sums up the found number of improvements from the runs that were previously presented in table 7.4.

As table 7.5 shows, the highest number of improvements (10) was found by the combination of the operators *remove related* and *greedy insertion each single customer complex forbidden*. Combining *remove related* with *greedy insertion each single customer simple perturbation* was very successful as well. The least successful operator combinations are *random DC removal* and *greedy insertion each customer simple perturbation*, *random customer removal* and *greedy insertion each single customer complex perturbation* as well as *worst customer removal complex* and *greedy insertion each single customer complex perturbation* and finally *worst removal from each DC simple* and *greedy insertion each single customer simple perturbation*.

With regard to the destroy operators, *random customer removal* seems to be the weakest with only 21 improvements found in total. *Remove related* found almost twice as many improvements with a total of 39 and was therefore the most successful destroy operator.

The performance of the repair operators seems more balanced. The improvements found here range from 25 (found by *greedy insertion each single customer simple perturbation*) and 37, which were found by *greedy insertion each single customer forbidden*.

Generally speaking, even though some operator combinations seem to be more efficient at finding improvements than others, none of them outperforms others sufficiently to fix that combination.

Throughout the performed tests, it happened frequently that the same operator was repeatedly successful in a specific run. This is due to the fact that its score gets higher for each found improvement and for this reason the operator is used more often in future runs.

Finally, this performance evaluation of all operators also helps to draw some conclusions about the success of the two different measures of distance, the simple and the complex calculations. As already mentioned, the simple insertion costs as well as the simple removal gain are represented by the distance or travel costs from a customer to the DC it is currently assigned to. The destroy operators working with simple removal gain calculations, which are *worst removal each single customer simple* and *worst DC removal simple*, in total found 68 improvements. The complex versions of the same operators, using the sum of distances of the customer to all other customers assigned to the same DC as a measure, found 66 improvements. This indicates that both distance measures are equally effective.

Regarding the repair operators, three operators use the simple distance calculation, namely *greedy insertion each single customer simple*, *greedy insertion each single customer simple perturbation* and *greedy insertion each single customer simple forbidden*. Together they found 91 improvements. Their versions with the complex distance calculation found 91 improvements as well, which strengthens the statements that both distance measures are efficient at finding improvements. For this reason, even though there might be some minor performance differences, we keep all the presented operators for future runs for a more diverse operator set.

operator	gr_cust_s	gr_cust_c	gr_cust_s_p	gr_cust_c_p	gr_cust_s_f	gr_cust_c_f	regret_s	total
rand_cust	2	3	3	1	4	2	6	21
rand_DC	3	3	1	3	8	4	3	25
worst_cust_s	8	8	2	5	3	3	2	31
worst_cust_c	6	2	7	1	6	8	3	33
worst_DC_s	9	4	1	7	6	4	6	37
worst_DC_c	4	7	2	4	6	2	8	33
remove_rel	4	3	9	7	4	10	2	39
total	36	30	25	28	37	33	30	219

Table 7.5: Number of improvements found by different operators and operator combinations in the runs from table 7.4 with 20 iterations as a stopping criterion for instances N400-Gs-Tr-U, N400-Gl-Tr-U and N400-Gl-Tbl-U and for 5, 10 and 20 iterations for instance N400-Gs-Tbl-U.

The short names of the operators used in table 7.5 are explained in the following overview:

- **rand_cust:** Random customer removal
- **rand_DC:** Random DC removal
- **worst_cust_s:** Worst customer removal simple
- **worst_cust_c:** Worst customer removal complex
- **worst_DC_s:** Worst removal from each DC simple
- **worst_DC_c:** Worst removal from each DC complex
- **remove_rel:** Remove related customer
- **gr_cust_s:** Greedy insertion each single customer simple
- **gr_cust_c:** Greedy insertion each single customer complex
- **gr_cust_s_p:** Greedy insertion each single customer simple perturbation
- **gr_cust_c_p:** Greedy insertion each single customer complex perturbation
- **gr_cust_s_f:** Greedy insertion each single customer simple forbidden
- **gr_cust_c_f:** Greedy insertion each single customer complex forbidden
- **regret_s:** Regret insertion simple

7.2 Instance Set 2

As mentioned in section 6, the second instance set is a new random instance. While most parameters do not change compared to set 1, we assume that the transportation costs are generally higher because of bad street quality, eventually caused by landslides or mud on the road, heavy rain in some areas or the slow speed of the used transportation vehicles. The higher costs also are due to the fact that deterioration is now also dependent on the route length or the transportation duration. As in previous runs, we again perform all the following runs on the 8 different versions of the instance. Each set of runs starts with a different starting solution and shows the cost improvement as well as the runtime average over a total of 5 runs per version. For this instance, the main difference is that the random costs of an arc, so its distance or travel time, is six times higher compared to the first instance set. Also, a different seed is used for the random instance generation.

Table 7.6 shows the first results. All the here performed runs are started with the distance based starting customer assignment from 7.1.1.

Generally speaking, the minimum (min % imp), maximum (max % imp) as well as the average (avg % imp) percentage improvement found by the ALNS-CCA is much higher for the large instance versions than for the small one. This seems logical since the number of customers, that can be reallocated, is very limited in the small instance set.

On average, the ALNS-CCA was able to improve the solution costs of between 22 and 27 percent for the large instance versions compared to the distance based starting

Data Set	min % imp	max % imp	avg % imp	min rt	max rt	avg rt
N400-Gs-Tbl-LM	20.8	24.9	22.5	61	166	104
N400-Gs-Tr-LM	23.3	28.2	25.6	43	65	50
N400-Gl-Tr-LM	23.4	28.7	26.5	35	55	46
N400-Gl-Tbl-LM	22.0	31.5	27.0	48	86	59
N50-Gs-Tbl-LM	1.4	3.7	2.2	6	8	7
N50-Gs-Tr-LM	1.2	2.6	1.6	6	8	7
N50-Gl-Tr-LM	2.3	3.5	2.8	6	8	7
N50-Gl-Tbl-LM	2.7	5.2	3.2	5	8	7

Table 7.6: Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 2. The starting assignment is created by the distance based algorithm from 7.1.1.

assignment from Prader (2022). The average runtime here was of between 46 (for the large instance with large grid size and a random terminal position) and 104 seconds (for the large instance version with a small grid and the terminal at the bottom left).

On the other hand, all runs with the small instance version took around 7 seconds but the found improvement only ranges between 1.6 and 3.2 percent on average.

In the next step, the same runs are repeated starting from the random starting customer assignment. Of course we would expect a higher percentage improvement found by the algorithm in this case. As table 7.7 shows, this expectation is also fulfilled. For all large instances the total costs could be improved of around 50 percent just by changing the customer assignment. And even for the small versions between 19 and 37 percent of the costs could be saved by the application of the ALNS-CCA on average. The runtimes are similar to ones of the previous runs.

Data Set	min % imp	max % imp	avg % imp	min rt	max rt	avg rt
N400-Gs-Tbl-LM	44.9	49.3	47.9	111	169	130
N400-Gs-Tr-LM	46.9	52.5	49.3	46	63	57
N400-Gl-Tr-LM	30.8	55.0	48.3	53	85	66
N400-Gl-Tbl-LM	51.1	54.4	53.1	62	98	80
N50-Gs-Tbl-LM	14.2	22.0	19.3	9	12	11
N50-Gs-Tr-LM	21.9	23.1	22.6	9	14	11
N50-Gl-Tr-LM	33.5	39.6	36.7	13	16	15
N50-Gl-Tbl-LM	31.4	34.7	32.5	9	15	11

Table 7.7: Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 2. The starting assignment is created by the random algorithm.

Finally, the same runs are repeated beginning from the regret based starting customer

assignment. The results over five runs are presented in table 7.8. It shows clearly, that for the current instance set the regret algorithm to generate a starting assignment is the most successful one, since the improvement that the algorithm could find is considerably smaller than for the two previously tested algorithms, which means that the starting solution was already relatively cost effective. To be precise, the costs for the large instance sets could only be improved of 12 to 15 percent and the improvements found for the small instance set are all below 3 percent.

Data Set	min % imp	max % imp	avg % imp	min rt	max rt	avg rt
N400-Gs-Tbl-LM	10.6	14.5	13.1	68	119	88
N400-Gs-Tr-LM	10.0	13.5	11.7	32	44	37
N400-Gl-Tr-LM	12.7	16.6	13.6	37	51	43
N400-Gl-Tbl-LM	11.1	17.9	15.0	59	82	71
N50-Gs-Tbl-LM	0.9	3.7	1.7	7	12	9
N50-Gs-Tr-LM	1.8	3.2	2.2	8	11	9
N50-Gl-Tr-LM	1.2	4.0	2.9	6	10	7
N50-Gl-Tbl-LM	1.2	4.8	2.5	7	10	9

Table 7.8: Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 2. The starting assignment is created by the regret based algorithm.

7.3 Instance Set 3

Figure 7.2 shows a typical instance from instance set 2. Here, the customer locations are generated randomly but they are spread more or less equally over the grid since a random generator based on a uniform distribution is used.

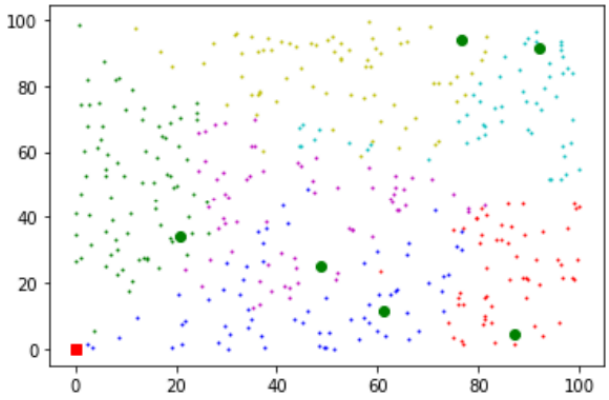

Figure 7.2: A typical instance of instance set 2 after the application of the ALNS-CCA.

Instance set 3 mainly changes the way how customers are spread over the grid. Now we assume that customers live in small villages or groups of houses, which is common especially on the countryside. For all the here tested instances, we will only consider a case with a large grid size of 100 times 100 units of measure. The reason is that with this instance we mainly want to find out how the different layout influences the customer clustering while changing the grid size changes the cost relations but not the optimal customer allocation. The different versions mainly differ in the number and size of the customer groups or villages. Again we will start from all three starting customer assignments and see how much improvement is possible if the algorithm is stopped at 10 iterations without improvement.

Data Set	min % imp	max % imp	avg % imp	min rt	max rt	avg rt
N400-Sl-RaA-V6	31.5	56.7	46.3	186	301	237
N400-Sl-ClA-V6	10.6	26.8	20.6	356	657	506
N400-Sl-RgA-V6	11.8	20.8	16.1	346	784	453
N400-Sm-RaA-V10	39.2	58.3	50.5	89	701	413
N400-Sm-ClA-V10	24.0	31.7	27.8	87	212	123
N400-Sm-RgA-V10	12.9	23.3	16.7	45	328	177
N400-Ss-RaA-V15	43.6	57.4	50.9	57	84	68
N400-Ss-ClA-V15	10.2	17.3	12.3	33	86	52
N400-Ss-RgA-V15	12.7	22.6	19.2	44	69	56

Table 7.9: Improvement found over 5 runs by the ALNS-CCA with varying numbers and sizes of villages.

The instance descriptions from table 7.9 are explained in more detail in the following overview:

- **Sl**: village size (S) large (l)
- **Sm**: village size (S) medium (m)
- **Ss**: village size (S) small (s)
- **RaA**: Random (Ra) starting customer assignment (A)
- **ClA**: Closest DC (Cl) starting customer assignment (A)
- **RgA**: Regret (Rg) starting customer assignment (A)
- **V6**: Number of villages (V) around 7
- **V10**: Number of villages (V) around 10
- **V15**: Number of villages (V) around 15

As table 7.9 shows, the runtime is considerably longer the less equally the customers are spread over the grid. Analysing the output in more detail leads to the conclusion, that the main reason for this is that the optimal solution of the first level seems more difficult when the customer are not spread uniformly. The algorithm could be sped up easily by stopping the solution of the first level at an earlier point, as two or five percent from optimal. However, for the sake of these tests we keep the optimality criterion.

Not surprisingly, the average percentage improvements found starting from the random customer assignment are always the highest since this is the weakest starting solution. For all three village sizes the improvement found here is of around 50 percent. The average improvement found compared to the closest DC assignment ranges between 12 and 28 percent and, therefore, is similar to the one found from the regret starting assignment, where it is of between 16 and 19 percent on average. If the distance based starting allocation or the regret allocation is more successful as a starting solution seems to depend on the instance version.

Image 7.3 shows one found final customer assignments on the first instance from set 3. This instance has around six large villages, even though there are some customers that live between the villages and the structure is relatively scattered. It also shows that the DC on the bottom left gets a higher number of customers assigned than the others because its position seems to be the most favourable. The algorithm therefore leads to a solution where the capacities of this DC are taken up more than the ones of the other centers. The two centers on the bottom right, on the other hand, get a relatively low number of customers assigned.

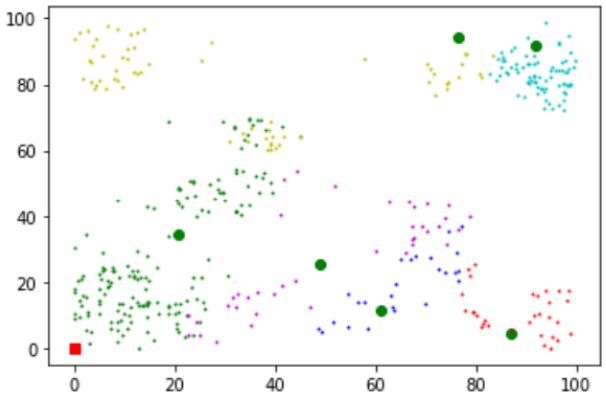

Figure 7.3: One of the found final customer assignments on version 1 from instance set 3.

Next, image 7.4 shows one example of a found final customer assignment on the second instance from set 3. The villages here are smaller and more numerous than in the previous instance. Even though the boarders between the villages blur because some of them touch each other, around 10 clusters of houses are visible here. The DC at the bottom left gets the highest number of customers assigned to while the DC in the bottom middle and the two DCs at the bottom right deliver a relatively small number of customer in this solution.

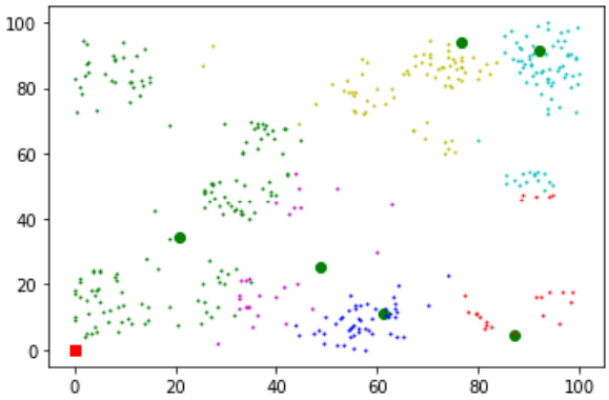

Figure 7.4: One of the found final customer assignments on version 2 from instance set 3.

Finally, image 7.5 shows one of the found final customer assignments on instance three from set 3. This last instance consists of many small villages. Some of them are clearly distinct from others while others grew together.

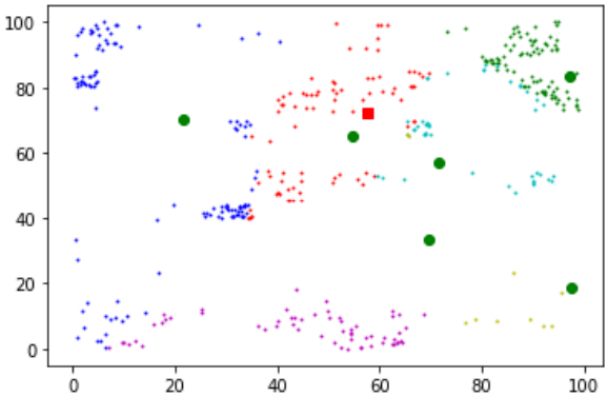

Figure 7.5: One of the found final customer assignments on version 3 from instance set 3.

7.4 Instance Set 4

As described in section 6, the fourth instance set is again generated randomly with 400 customers and 6 DCs. Small instances again consist of 50 customers and 3 DCs. A new seed is used for the numpy random generator which leads to a different instance compared to the previous set. What we will test now is the impact of suddenly changing street conditions after the transportation of a new period was already planned.

Let us assume that we have given distances between nodes and given travel costs for each arc. With this information, the problem is solved and the customers are assigned to DCs as done previously. However, just before the delivery period begins, heavy rain

makes some streets much more costly, for instance dirt roads which become very muddy. Other streets, however, are not affected because the rain was less strong in the area or because the street condition is better or, eventually, a better canalization system is in place. A similar situation could happen after natural disasters like floods, tsunamis, earthquakes, landslides etc. In instance set 4 we first solve a transportation problem like in the previous cases. Before the week starts, the costs of some streets will increase. In our instance it will be the costs of the first half of the edges and, as a simplification, we just double them. After this, we recalculate the total costs of the found solution, which at this point is expected to not be optimal anymore. The ALNS-CCA will finally be applied on the solution again to find a better customer assignment with the new cost situation. Like this we attempt to prove that the ALNS-CCA is an efficient method to react quickly to changing street conditions. The customers are again spread uniformly over the grid as in instance sets 1 and 2.

Data Set	total costs before	total costs after	total costs ALNS-CCA
N400-Gs-Tbl-U	714.7	1091.5	1029.0
N400-Gs-Tr-U	866.6	1268.2	1153.7
N400-Gl-Tr-U	6711.1	10 255.2	9664.2
N400-Gl-Tbl-U	6024.7	10 616.0	9739.6
N50-Gs-Tbl-U	1547.1	1646.3	1632.5
N50-Gs-Tr-U	1748.7	1848.6	1815.5
N50-Gl-Tr-U	2819.0	4623.1	4231.2
N50-Gl-Tbl-U	3416.3	4421.1	4251.6

Table 7.10: Total costs of the best found solution with the original costs (total costs before) compared to the costs of that solution after a cost change (total costs after). The last column of the table presents the total costs of a solution with the new costs that is again improved by the ALNS-CCA (total costs ALNS-CCA).

In all our eight known instance versions the total costs before the sudden cost change are given. As soon as the costs of the first half of the arcs are doubled, these costs of course increase substantially. In some cases, as for instance N50-Gs-Tr-U, the changed cost conditions do not affect the solution too much. However, in other cases, especially instance N50-Gl-Tr-U, the supplier is more unfortunate and the original solution is now almost twice as costly as it used to be. For all instance versions the anew application of the ALNS-CCA helps to decrease those costs again, at least to a certain extend.

The final customer assignment on a typical large instance from set 4 is shown in figure 7.6

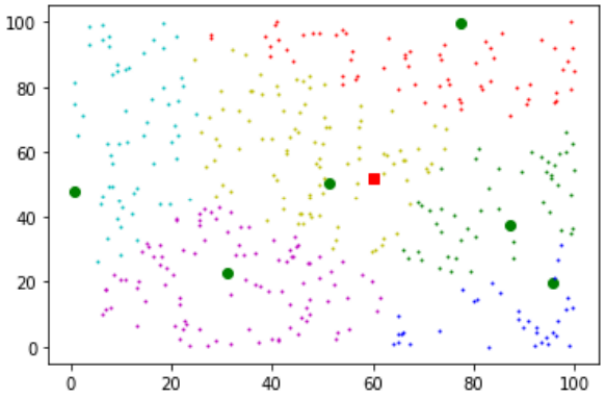

Figure 7.6: One of the found final customer assignments on version 4 of instance set 4.

Finally, figure 7.7 shows the final customer assignment for the small instance from instance set 4 with the terminal at the bottom left.

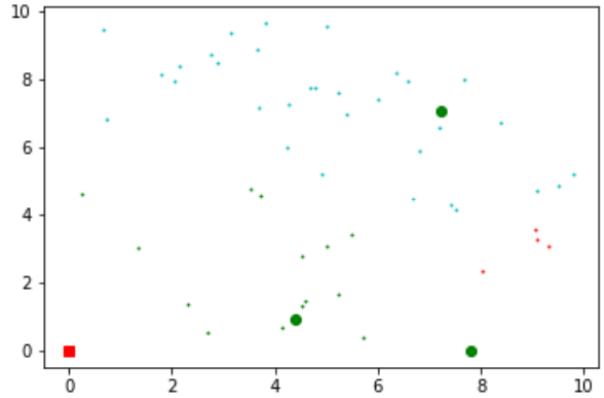

Figure 7.7: One of the found final customer assignments on the small version of instance set 4 with the terminal at the bottom left of the grid.

7.5 Instance Set 5

The fifth and last instance set deals with an extreme example of changing conditions: it simulates a case where, in a transportation problem similar to the previous ones, one DC must close for the whole planning horizon. This can be because of damage caused through a natural disaster, as an earthquake, or because the streets leading to it are completely flooded. Big forest fires due to climate change or the burning season in countries as Myanmar, Thailand or Lao can also lead to such a problem. In general, the instance is very similar to the ones used in set 1 and 2, only that we have one less DC which means the customers previously assigned to this DC must be redistributed to the remaining ones. This is a case where having an efficient algorithm to change customer allocations to DCs, as the ALNS-CCA, is advantageous. Such a setting, of course, requires that the capacity of the other DCs and of their vehicles is sufficient to cover the demand of the inactive DC as well. We assume that this is the case if the customer assignment is done effectively. Since the ALNS-CCA interrupts iterations that lead to exceeding capacities, it is guaranteed that each solution presented is feasible.

A typical large instance of instance set five looks as presented in figure 7.8 We assume that there was another DC at the middle left of the grid that is currently inactive, which forces the other DCs to take over and use more of their available capacities. Not surprisingly, most customers of the inactive DC are now assigned to its two closest DCs while some of the ones that are further away have less assigned customers. However, it still depends strongly on the capacity restriction to which extend this is possible.

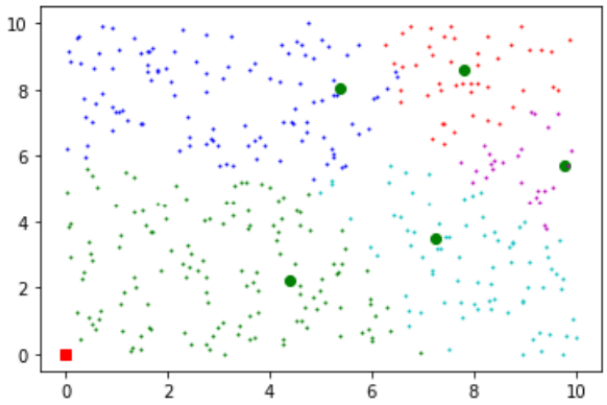

Figure 7.8: One of the found final customer assignments on the large version of instance set 5 with the terminal at the bottom left of the grid.

The small instances now have only two DCs left instead of three. This looks as presented in figure 7.9.

The tests performed on this instance will be the same as for instance set two. We will again start from all three different starting customer assignments and run the code 5 times with 10 iterations without improvement found as a stopping criterion. Again, we will report the minimum and maximum improvement found over these runs as well as the

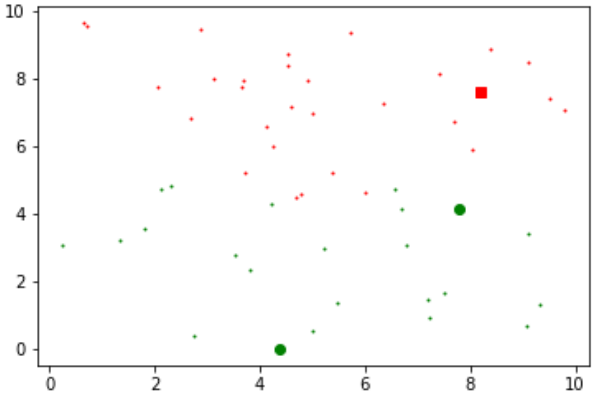

Figure 7.9: One of the found final customer assignments on the small version of instance set 5 with the terminal at a random position.

average. Also the minimum, maximum and average runtime will be reported.

Table 7.11 shows the results for the runs started from the distance based customer assignment. The found improvements in this case are relatively high for this starting allocation with 24 to 31 percent for the large instance versions and 2 to 9 percent on the small ones.

Data Set	min % imp	max % imp	avg % imp	min rt	max rt	avg rt
N400-Gs-Tbl-DZ	21.8	26.1	24.7	62	90	80
N400-Gs-Tr-DZ	22.8	26.0	24.2	55	74	66
N400-Gl-Tr-DZ	24.3	32.7	29.9	59	104	92
N400-Gl-Tbl-DZ	30.7	31.8	31.1	105	171	139
N50-Gs-Tbl-DZ	1.7	1.9	1.8	6	9	8
N50-Gs-Tr-DZ	1.4	2.7	2.3	7	11	9
N50-Gl-Tr-DZ	6.2	11.8	8.8	8	10	9
N50-Gl-Tbl-DZ	3.4	9.4	7.4	6	9	7

Table 7.11: Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 5. The starting assignment is created by the distance based algorithm from 7.1.1.

The same runs are next repeated starting from the random customer assignment. Table 7.12 shows the performance of the ALNS-CCA on this setting. Not surprisingly, the percentage improvement found here is again generally higher than for the one with the stronger starting assignment. The average improvement found on the large instance versions ranges between 38 and 50 percent. For the small instances improvements of 2 to 14 percent were found on average.

Data Set	min % imp	max % imp	avg % imp	min rt	max rt	avg rt
N400-Gs-Tbl-DZ	27.2	43.7	37.7	56	105	78
N400-Gs-Tr-DZ	40.4	42.7	42.0	46	65	53
N400-Gl-Tr-DZ	45.4	51.0	49.8	55	83	64
N400-Gl-Tbl-DZ	40.4	49.7	47.4	148	293	187
N50-Gs-Tbl-DZ	0.5	2.8	2.0	7	9	8
N50-Gs-Tr-DZ	1.4	4.3	3.4	7	9	7
N50-Gl-Tr-DZ	10.2	18.2	13.8	8	11	10
N50-Gl-Tbl-DZ	11.6	12.3	11.8	5	9	7

Table 7.12: Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 5. The starting assignment is created by the random customer assignment algorithm.

Finally, we repeat the same runs starting from the regret customer assignment. The algorithm performance is presented in table 7.13. As this table shows, the regret starting algorithm again performed best since the found improvements are lower compared to the other two starting algorithms. In general, the runtimes do not differ strongly depending on the chosen starting assignment generation.

Data Set	min % imp	max % imp	avg % imp	min rt	max rt	avg rt
N400-Gs-Tbl-DZ	14.0	17.1	15.3	45	59	53
N400-Gs-Tr-DZ	11.6	15.1	13.8	32	45	37
N400-Gl-Tr-DZ	13.4	18.7	17.1	46	52	49
N400-Gl-Tbl-DZ	13.1	19.2	16.2	78	108	92
N50-Gs-Tbl-DZ	0.2	0.8	0.6	5	8	6
N50-Gs-Tr-DZ	2.0	2.2	2.1	6	7	7
N50-Gl-Tr-DZ	6.3	9.1	7.9	6	8	7
N50-Gl-Tbl-DZ	1.6	3.9	3.0	5	7	6

Table 7.13: Improvement found over 5 runs by the ALNS-CCA with 10 iterations without improvement as a stopping criterion on the 8 versions of instance set 5. The starting assignment is created by the regret customer assignment algorithm.

7.6 Efficient combination of ALNS-CCA and ALNS-DP

The probably most intuitive way to combine the two improvement algorithms, namely the ALNS-CCA and the ALNS-DP, is by using them successively. It seems reasonable to use the ALNS-CCA first to find a good customer assignment and to improve the solution

found like this further by changing some delivery patterns. The following section will, first, try to find a way to speed up the feedback loop and the ALNS-DP and, after that, test how well the two improvement algorithms operate in combination.

In the following experiments we will run the ALNS-CCA first and add one run of the ALNS-DP with 100 iterations for additional improvements.

7.6.1 Improvements and tests of the ALNS-DP

Since reducing the number of improvement iterations of the ALNS-2LR in the generation of the total solution was very successful for the ALNS-CCA, we tried the same with the ALNS-DP. To be precise, whenever the ALNS-2LR is used in the feedback loop from Prader (2022), we stop it at 10 iterations instead of 1000.

Again, like this the total solution is already relatively close to the optimal and our experiments show, that this is sufficient to find real improvements. Additionally, the algorithm like this operates considerably faster than it did in Prader (2022).

We use this updated version of the ALNS-DP and the feedback loop to see how good the found improvements over 100 iterations of the ALNS-DP are compared to the best found solution after the application of the ALNS-CCA. We decided to stop the run at 100 iterations of the feedback loop because in our tests, running it with 1000 iterations, did not bring a considerable improvement compared to it. In most cases almost all improvements were found in the first 100 iterations.

For these tests the urban transportation instance set, so instance set 1, was used. The starting customer assignment for the ALNS-CCA was generated by the regret based algorithm since it proved to be the most successful for most instances.

Data Set	c after ALNS-CCA	c after ALNS-DP	# imp.	rt. ALNS-DP
N400-Gs-Tbl-U	1129.5	1082.7	14	53
N400-Gs-Tr-U	940.8	894.6	13	101
N400-Gl-Tr-U	6884.3	6384.5	7	72
N400-Gl-Tbl-U	7548.0	6977.2	10	69

Table 7.14: Results of a run with the following steps: first the ALNS-CCA is used on the starting solution. The found total costs are reported (c after ALNS-CCA) and the found customer assignment is passed on to the ALNS-DP that additionally improves the delivery patterns with 100 iterations as a stopping criterion. The costs of the best found solution here are again reported (c after ALNS-DP), together with the number of improvements found by the feedback loop (# imp.) and the runtime of the ALNS-DP (rt. ALNS-DP) with the new settings.

The tests in table 7.14 show clearly how much faster the performance of the adapted feedback loop is compared to the version in Prader (2022), where many runs had to be stopped at the first improvement due to the high computational costs. It also proves that having some flexibility regarding the delivery days does actually really make a difference

and helps saving fuel and waste, even compared to a solution with optimized customer assignments.

7.6.2 Conclusion on the ALNS-CCA and ALNS-DP

There is one main advantage in having two separate improvement algorithms working on different parts of the solution: they can be applied independently. As the experiments in this computational study showed, both algorithms are able to find considerable improvements to the solution. Even though the starting solution, consisting of the optimal solution of the first level and the heuristic solution of the second level, was already good enough to be accepted by other peers, like Rohmer et al. (2019), we were able to improve such a solution in two different ways.

In a case where we are not forced to visit a fixed set of customers from each DC, making the customer assignment flexible makes the routing more efficient, cost saving and food safety might be enhanced.

If customers additionally allow the company to choose their delivery dates, the amount of waste produced in the supply chain can be reduced even further and even more transportation and storage costs can be safed.

To sum up, the computational study clearly proves that the proposed method generates a complete solution to a complex mathematical problem. We also found two effective ways to improve it, which can be used independent from each other or in combination, depending on the application.

8 Conclusion

Providing the world's population with food is an extremely impactful but also very complex task, since it includes several considerations. Guaranteeing that a sufficient amount of groceries is available is only a part of the solution. Since those items are perishable, it is necessary to make sure that they are still fresh, nutritious and safe to eat as soon as they reach the end consumer. Additionally, the way how these products get to the customer is usually connected to the emission of GHGs and other negative externalities. Also, avoidable costs of inefficient transportation and storage as well as wasted food items make those goods less affordable.

Alarmingly, the GFSI, which evaluates food provision based on factors such as affordability, sustainability and food quality and safety, has seen a decline in recent years (UNCCD, 2022). For instance, food production still accounts for over one quarter of the worldwide GHG emission and more than half of the habitable land on our planet is used for agriculture. Since the global population continues to grow and the consequences of climate change become more and more noticable, addressing this problem becomes increasingly important. (Ritchie et al., 2022)

All the mentioned factors are relevant in inventory routing systems for food delivery, which exist all over the world. In the research for this thesis, food supply systems in different Asian countries on the lower-middle income level were observed. We found that the conditions vary significantly due to climate, geography and street quality, as well as the mode of transportation that is chosen. Even culture has a strong impact on what is delivered and how. For all those reasons, different countries also face distinct issues regarding food security. As mentioned in the literature review, all observed countries perform relatively poorly in the sustainability category. For example, Lao has room for improvement in all categories of the GFSI, while Nepal still struggles with low standards of food quality and safety for end customers.

Given the diversity of these problems regarding costs and circumstances, a flexible solution method that is adaptable to the given conditions in different contexts is required. Finding and implementing such a method was the objective of this master's thesis.

We aimed at enhancing the existing solution method proposed by Prader (2022) and increase its adaptability to diverse supply chains in various countries worldwide. To achieve this, a novel improvement algorithm, called ALNS-CCA, was developed. The ALNS-CCA algorithm strives to minimize costs associated with waste, transportation, and storage of perishable items by optimizing customer allocation to DCs. In our tests, the ALNS-CCA found considerable improvements compared to the starting customer assignment generated by three developed customer allocation algorithms. It proved to find a cost-effective assignment of customers that guarantees that no capacity restriction is exceeded. By shortening the delivery duration and route lengths it contributes to a

reduction of food waste and food insecurity as well as general (environmental) costs of the supply chain.

While the problem is, as in Prader (2022), formulated as a 2E-IRP with mandatory consolidations at the DCs and considerations for the perishability of food items, the application of this method is expanded to a broader context and tested on more diverse instances. To be precise, we compared the performance of the algorithm on urban transportation instances with the delivery of small villages on the countryside, varied the size of the grid and the location of the major terminal. We also considered transportation on streets of poor or changing quality and relocated customers after natural disasters.

Additionally, we found a way to enhance the improvement method from Prader (2022), a feedback loop that reduces the total costs by choosing the patterns in which customers are delivered.

After testing both improvement strategies extensively, we came to the conclusion that combining them as well as using them independently from each other is very effective at reducing the total costs of the problem. Both tested strategies help enhance food safety by ensuring that no deteriorated or contaminated groceries reach the final consumer. They also contribute to the reduction of food waste and GHG emissions, thereby improving the environmental footprint of the supply chain. By minimizing the costs of storage and transportation, as well as the amount of food produced, transported, and stored without being consumed, the affordability of food items is enhanced. This is particularly important for countries with low income levels.

Even though the present research proved to provide several new ways to make food more affordable and safe and its supply less harmful to the environment, there are some limitations. Of course we are aware that not every type of food supply system can flexibly reschedule delivery dates for final customers while others might require a fixed customer assignment to DCs. Prader (2022) as well as Rohmer et al. (2019) presented methods to solve such problems efficiently. The two improvement algorithms are, however, an efficient extension if further enhancements are needed and they are flexibly addable and usable if required.

While we tested the new methods only on problems regarding food provision, lower-middle income countries often also struggle with waste collection. Adapting the model and the method to such systems could be an interesting field for future research. Since waste of all types is currently burned in many of the countries we visited, or abandoned in nature where it can lead to problems as contaminated water sources, this adaptation would surely add value.

In conclusion, this research provides a significant step forward in addressing the complex challenges of global food supply systems. The findings and methods presented here contribute to improving food affordability, safety, and environmental sustainability, thereby offering valuable insights for policymakers, practitioners, and researchers.

9 References

Azadeh, A., Elahi, S., Farahani, M. H., & Nasirian, B. (2017). A genetic algorithm-Taguchi based approach to inventory routing problem of a single perishable product with transshipment. *Computers & Industrial Engineering*, *104*, 124-133.

Charaf, S., Taş, D., Flapper, S. D., Van Woensel, T. (2022). A Branch-and-Price Algorithm for the Two-Echelon Inventory-Routing Problem. arXiv preprint arXiv:2206.12316.

Coelho, L. C., Cordeau, J. F., & Laporte, G. (2014). Thirty years of inventory routing. *Transportation Science*, *48*(1), 1-19.

Cuda, R., Guastaroba, G., & Speranza, M. G. (2015). A survey on two-echelon routing problems. *Computers & Operations Research*, *55*, 185-199.

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management science*, *6*(1), 80-91.

Ehmke, J. F., Campbell, A. M., & Thomas, B. W. (2018). Optimizing for total costs in vehicle routing in urban areas. *Transportation Research Part E: Logistics and Transportation Review*, *116*, 242-265.

Farias, K., Hadj-Hamou, K., & Yugma, C. (2020). Model and exact solution for a two-echelon inventory routing problem. *International Journal of Production Research*, 1-24.

Federgruen, A., Prastacos, G., & Zipkin, P. H. (1986). An allocation and distribution model for perishable products. *Operations research*, *34*(1), 75-82.

Gapminder. Income Level 2. Retrieved February 13, 2023, from https://www.gapminder.org/fw/income-levels/income-level-2.

Guimarães, T. A., Coelho, L. C., Schenekemberg, C. M., & Scarpin, C. T. (2019). The two-echelon multi-depot inventory-routing problem. *Computers & Operations Research*, *101*, 220-233.

Hemmelmayr, V. C., Cordeau, J. F., Crainic, T. G. (2012). An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. *Computers operations research*, *39*(12), 3215-3228.

Hiassat, A., & Diabat, A. (2011). A location-inventory-routing-problem with perishable products. In *41st International Conference on Computers and Industrial Engineering 2011* (pp. 130-135).

Hiassat, A., Diabat, A., & Rahwan, I. (2017). A genetic algorithm approach for location-inventory-routing problem with perishable products. *Journal of manufacturing systems*, *42*, 93-103.

Nahmias, S. (2011). *Perishable inventory systems* (Vol. 160). Springer Science & Business Media.

Onggo, B. S., Panadero, J., Corlu, C. G., Juan, A. A. (2019). Agri-food supply chains

with stochastic demands: A multi-period inventory routing problem with perishable products. *Simulation Modelling Practice and Theory*, *97*, 101970.

Prader, R. (2022). Urban Two Echelon Inventory Routing Problem for perishable goods with a waste reduction objective. [Master's thesis, University of Vienna]

Rabbani, M., Farshbaf-Geranmayeh, A., & Haghjoo, N. (2016). Vehicle routing problem with considering multi-middle depots for perishable food delivery. *Uncertain Supply Chain Management*, *4*(3), 171-182.

Rahimi, M., Baboli, A., & Rekik, Y. (2017). Multi-objective inventory routing problem: A stochastic model to consider profit, service level and green criteria. *Transportation Research Part E: Logistics and Transportation Review*, *101*, 59-83.

Ritchie, H., Rosado, P. Roser, M. (2022). Environmental Impacts of Food Production. Retrieved May 12, 2023, from https://ourworldindata.org/environmental-impacts-of-food

Rohmer, S. U. K., Claassen, G. D. H., & Laporte, G. (2019). A two-echelon inventory routing problem for perishable products. *Computers & Operations Research*, *107*, 156-172.

Ropke, S., & Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, *40*(4), 455-472.

Schenekemberg, C. M., Scarpin, C. T., Pécora Jr, J. E., Guimarães, T. A., & Coelho, L. C. (2020). The two-echelon inventory-routing problem with fleet management. *Computers & Operations Research*, *121*, 104944.

Schenekemberg, C. M., Scarpin, C. T., Pecora Jr, J. E., Guimarães, T. A., Coelho, L. C. (2021). The two-echelon production-routing problem. European Journal of Operational Research, 288(2), 436-449.

Soysal, M., Bloemhof-Ruwaard, J. M., Haijema, R., & van der Vorst, J. G. (2015). Modeling an inventory routing problem for perishable products with environmental considerations and demand uncertainty. *International Journal of Production Economics*, *164*, 118-133.

UNCCD. (2022). The Global Food Security index. Retrieved February 15, 2023, from https://www.unccd.int/resources/knowledge-sharing-system/global-food-security-index

United Nations. (2022). Food Loss and Waste Reduction. Retrieved February 13, 2023, from https://www.un.org/en/observances/end-food-waste-day

Violi, A., Laganá, D., Paradiso, R. (2020). The inventory routing problem under uncertainty with perishable products: an application in the agri-food supply chain. *Soft Computing*, *24*(18), 13725-13740.

World Health Organization. (2022). WHO global strategy for food safety 2022–2030: towards stronger food safety systems and global cooperation: executive summary. World Health Organization. https://apps.who.int/iris/handle/10665/364638. License: CC BY-NC-SA 3.0 IGO

Zhao, Q. H., Chen, S., & Zang, C. X. (2008). Model and algorithm for inventory/routing decision in a three-echelon logistics system. *European Journal of Operational Research*, *191*(3), 623-635.