Optimising storage assignment, order picker routing, and order batching for an e-grocery fulfilment centre: an exact and heuristic approach

Master thesis Aerospace Engineering

Merel Sterk

Optimising storage assignment, order picker routing, and order batching for an e-grocery fulfilment centre: an exact and heuristic approach

Master thesis Aerospace Engineering

by

Merel Sterk

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on December 19, 2022.

Student number:	4447182	
Project duration:	January 2022 – December 2022	
Thesis committee:	Dr. ir. M.I. Gerritsma	TU Delft, Chair
	Dr. A. Bombelli	TU Delft, Supervisor
	ir. P.C. Roling	TU Delft, Examiner
	ir. F. Boekema	Picnic, Supervisor

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Acknowledgements

Hereby, I present my final thesis project to obtain my masters degree at the faculty of Aerospace Engineering of Delft University of Technology. Having started studying in Delft some years ago, I am both excited and sad that my time as a student here will come to an end. It is bittersweet since I have thoroughly enjoyed my time in Delft and at the University, however I am also ready and excited about the new challenges and opportunities ahead! During my thesis project at Picnic Technologies, I have had the chance to end my career as a student in a fast-paced and motivating environment, which I am very grateful for. So, before I encourage you to go on to read the results of my work, I would like to thank some key people that were all part of this project.

First and foremost, I would like to thank Alessandro. Your optimism, intelligence and critical assessments made sure that I was always prepared for the next steps of the thesis and kept me sharp throughout. Your knowledge of both MILP and ALNS was invaluable and for that I want to thank you. Also, I would like to thank Paul Roling for your contribution during all the milestone meetings and sharp eye when it comes to operations research. A thanks goes out to Marc Gerritsma as well, for being the chair of my thesis committee.

From Picnic, firstly I would like to thank Floris. Along with everybody else at Picnic, you really helped me with all your insights into the operations and your experience on the subject. Both the Fulfilment Analyst team and you continued to push me to think about the business implications of the project, new and innovative ideas, and about how we could improve our solutions. Next to that, I want to thank Jop. Having done most of our projects throughout the master together, it only made sense that we finished our master by working at the same place as well. You were always there to offer words of encouragement and to remind me of the fact that I was not alone in this endeavour! Last but not least, I want to express my gratitude to my family, who have made sure that they could always provide a listening ear whilst offering support and motivation. Let us celebrate together!

Merel Sterk
Amsterdam, December 2022

Contents

List of Figures

List of Tables

List of Abbreviations

ALNS	Adaptive Large Neighbourhood Search
BBD	Best-Before Date
COI	Cube-per-Order Index
DWH	Data Warehouse
FC	Fulfilment Centre
LNS	Large Neighbourhood Search
MILP	Mixed Integer Linear Programming
MSL	Multiple Storage Locations
NP	Non-deterministic Polynomial-time
OBP	Order Batching Problem
OPRP	Order Picker Routing Problem
SAP	Storage Assignment Problem
SKU	Stock Keeping Unit
SQL	Structured Query Language
TSP	Travelling Salesman Problem
WMS	Warehouse Management System

Introduction

Since 2015, Picnic Technologies has aimed at disrupting the traditional supermarket industry in The Netherlands and beyond its borders. With the demand for online grocery shopping ever growing, it is crucial to stay innovative and ahead of the traditional big players in this highly competitive market [2]. Since the e-grocery market is a small margin market, just like the offline grocery market, it is essential to save costs where possible [4]. The biggest portion of costs during the fulfilment process lies in order picking: the process of retrieving products from storage in response to a specific customer request [1]. Hence, saving costs in order picking can significantly reduce the costs of the overall supply chain. A novel way to try to reduce costs is by introducing Multiple Storage Locations (MSL): storing the same article, or stock keeping unit (SKU), at multiple locations throughout the pick circuit, such that it can be picked from multiple locations instead of only one. Combining this with more efficient routing of the order pickers, and better batching of orders increases the chance that order pickers only have to travel through part of the pick circuit. Having to traverse less distance with heavy pick carts saves time, which can be converted to cost savings. Next to that, the demand for picking from one storage location will be spread out over multiple locations for some SKUs. This might have a mitigating effect towards decreasing congestion in overcrowded pick circuits. To study if this reduction of cost takes place and if the effect is significant enough to implement MSL along with SKU allocation, order picker routing, and order batching into operations, both an exact model and a meta-heuristic model have been constructed. With the input of historic order data and information on the SKUs, the models can construct new layouts where SKUs are stored at multiple locations, albeit within the capacity limits of the building. For the exact model, Gurobi was used together with a Mixed Integer Linear Programming (MILP) formulation. This model was compared to the meta-heuristic Adaptive Large Neighbourhood Search (ALNS) model, which uses several destroy- and repair heuristics to modify the solution each time in search for the best outcome.

The goal of this master thesis is to obtain a degree in Aerospace Engineering for the track of Control & Operations, as well as to conduct a case study about the subject at Picnic Technologies. The research consists of two main parts; the MILP model and the ALNS model. The case study for Picnic was executed with both models, with the goal to reduce costs and increase efficiency. The goal of the MILP model is to set a benchmark for the ALNS model such that the performance of the ALNS model can be tested against exact solutions. Meanwhile, the ALNS model can handle larger instances in a shorter amount of time to ensure that business decisions can be taken based on the output from the model.

This report is comprised of three parts. In the first part, the scientific article can be found. In this article, the methodology is explained, the results are discussed and conclusions are drawn. Part II contains the literature study. This was done at the beginning of the research and consists of the problem description and the research question and its sub-questions, to guide the research. Furthermore, the state-of-the-art research from qualitative sources that were already available is discussed and comparisons between problems are made. Also, different solution methods for said problems are laid out. Finally, the third section contains all the supplementary work.

I

Scientific Paper

Optimising storage assignment, order picker routing, and order batching for an e-grocery fulfilment centre: an exact and heuristic approach

Merel Sterk

Delft University of Technology, Delft, The Netherlands

Abstract

With (e-)grocery retailers striving to increase their efficiency and subsequently reduce their costs, the opportunities that lie in optimising the order picking process of the supply chain is of key importance. The complex nature of assigning articles to an optimal storage location, having efficient routing of order pickers, and optimised grouping of orders calls for an integral approach to these problems. The objective of the research presented in this paper is to reduce the travelled distance of order pickers and subsequently the costs of the order picking process. Next to the integral approach, a new method of Multiple Storage Locations (MSL) is introduced. Historic order data of an e-grocery retailer is used, together with information on stock keeping units (SKUs), to implement SKU allocation with MSL possibility, routing, and batching into models. This ensures that the cost saving effects of these measures can be quantified. A benchmark Mixed Integer Linear Programming model (MILP) is developed and compared to a meta-heuristic Adaptive Large Neighbourhood Search model (ALNS) to determine how much travelled distance would be saved. The ALNS has multiple destroy- and repair heuristics, some of which are novel, that are specific to the problem at hand. The ALNS is able to handle bigger instances than the MILP, whilst ensuring quality of the solution. The MILP model outperforms the ALNS for small instances, however for large instances (instances of 400 orders or more) the ALNS performs, on average, 12.5% better whilst reducing computational time by 14.8%. Finally, areas of improvement are suggested in the ALNS model as well as other effects that should be studied when introducing MSL.

Keywords: *storage assignment problem, routing problem, batching problem, adaptive large neighbourhood search, e-grocery retailer, order picking, multiple storage locations.*

1 Introduction

Over the last decade, online purchasing has been on the rise, which is evident by the growing e-commerce sector [Alfonso et al. (2021)]. One of the interesting markets, and a late bloomer in this sense, is the online grocery market (e-grocery). The e-grocery market comprises purchasing and selling food products or grocery items online. The key difference between e-grocery retailers and traditional supermarkets is the order fulfilment process, which for e-grocery retailers takes place in the Fulfilment Centres (FCs). In the FCs, orders are assembled and there are many critical and supporting processes taking place to fulfil the orders in a timely manner. One of the critical processes is the order picking process: retrieving products from storage locations in response to a specific customer request. Since the (e-)grocery market is traditionally a low-margin industry, cost savings are of importance. Previous research has shown that order picking is estimated to make up 55% of the entire fulfilment costs [Füchtenhans et al. (2021)]. Hence, to cut down on costs, it is important to look at cost reductions within the order picking process. Cutting costs in this process can help to reduce the overall fulfilment costs significantly because of the large share of order picking costs. This area of research has gained a lot of interest over the years with the arrival of many FCs. There are a variety of studies dedicated to this topic.

Currently, there are two main types of order picking systems that exist: *parts-to-picker* systems and *picker-to-parts* systems. The former is often present in automated FCs and it entails that the part (i.e. item) is moved to the order picker, for example via conveyor belts. The latter system is used in manual FCs. For this research, the focus is on manual FCs and thus a picker-to-parts order picking system. In this system, order pickers traverse aisles in the pick circuit, to retrieve all the required items from the storage locations. The pick circuit consists of all storage locations of items, usually separated into different aisles. The efficiency of the order-picking process is mainly dependent on three factors: (i) the storage assignment, (ii) the routing of the order pickers, and (iii) the batching of orders [Dukic and Oluic (2007), Zhang et al. (2019)].

Which item, or Stock Keeping Unit (SKU), is stored at which location is decided before the order picking process commences. Assigning SKUs to storage locations is known as the Storage Allocation Problem (SAP). Deciding the optimal storage location for an SKU is both cardinal and complex, especially for e-grocery retailers since they can house over 10,000 different SKUs. The routing of the order pickers through the pick circuit can be decided by many different strategies, largely dependent on the order (type) and layout of the FC, which is known as the Order Picker Routing Problem (OPRP). Also, the batching of orders is an effective measure to cut down on time when executed properly, since an order picker can collect multiple orders simultaneously. Deciding which orders are batched together is known as the Order Batching Problem (OBP). In FCs, the batching of orders is done on pick carts. For e-grocery retailers, these pick carts contain several order totes (plastic crates in which SKUs of the order are assembled), where each order tote contains (part of) the SKUs of an order.

This research is taking a novel integral approach to three factors influencing the efficiency of the order picking process, to reduce the travel distance of order pickers. Next to that, Multiple Storage Locations (MSL) are introduced. MSL entails that an SKU does not have one storage location. Instead, the stock can be split over multiple locations throughout the order pick circuit. The idea of MSL is maximising the likelihood that the order pickers will only need to traverse a portion of the pick circuit. This might bring extra burden to other parts of the operational processes and therefore a limit is set to the maximum number of MSL. An illustrative example of how this would work is shown in Figure 1. Introducing MSL is not an entirely new concept within supply chain management, however, it is for e-grocery retailers. The main difference between e-grocery retailers and other e-commerce businesses is that the orders for e-grocery retailers are generally of much bigger size, i.e. contain more SKUs. The effect that MSL can have in combination with this bigger order size and an integral approach to the SAP, OBP and OPRP has not been widely studied yet and has not been adopted into practice as far as the author knows.

To solve these intertwined problems, a model is introduced that can concurrently solve the SAP without constraining an SKU to one storage location (i.e. having MSL), the OPRP, and the OBP. Combining these three problems and working towards an integral solution is paramount to the performance of the model. With the reduction of travelled distance, the order pickers will spend less time fulfilling an order. This entails that the costs to fulfil an order are reduced, since labour costs make up the biggest portion of the order picking process. This will make the e-grocery retailer that adapts its operations to the model competitive and can ensure that they have the edge over their competitors.

There are several methodologies to solve these problems and reach the goal of reduced travelled distance by order pickers. A commonly used approach is constructing a Mixed Integer Linear Programming (MILP) model. A MILP model ensures an optimal solution to the problem. However, due to the large and complex nature of the problems, considerable computational time is needed before an optimal solution is reached for instances that are interesting from a practical, operational perspective. To circumvent this problem, meta-heuristics are a valid alternative, as they might not find an optimal solution, yet they are generally capable of finding a near optimal solution within an acceptable time limit. The meta-heuristic that is proposed in this research is an Adaptive Large Neighbourhood Search (ALNS), which combines the strength of several other meta-heuristics into one algorithm. By inputting order data and retrieving a proposed storage location assignment, the reduction of distance can be computed when the routing and batching are considered as well.

This paper is organised as follows. In Section 2 a summary of the existing academic literature on order picking processes is given. Thereafter, a detailed description of the problem is given in Section 3. Next, Section 4 describes both the exact methodology and the meta-heuristic approach that were used. Afterwards, Section 5 describes the computational experiments that were executed. This leads to the results and analysis thereof that are shown in Section 6. Finally, Section 7 presents the conclusions of the research and areas for further research.

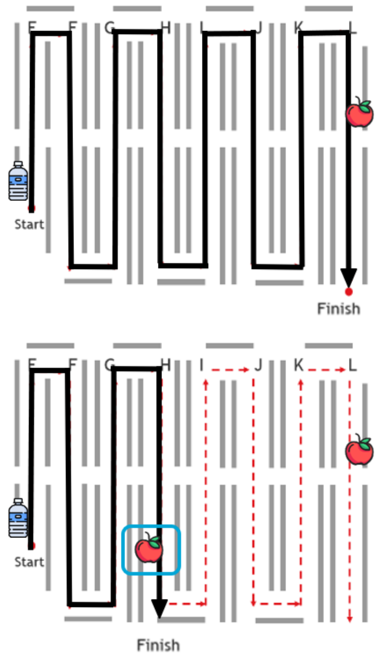


Figure 1: Illustrative example of how introducing MSL can reduce travelled distance. Top view of the order pick circuit with eight aisles labelled E-L. Both the water bottle and the apple are needed. In the situation on top, the full circuit needs to be traversed to pick both SKUs. In the situation on the bottom, the apple has MSL and only half of the pick circuit needs to be traversed.

2 Literature Review

Ensuring the most beneficial order picking practices in terms of travelled distance is a complex task. Over the years, many methods have been proposed to tackle this problem with many different focuses. In general, the order picking process can be divided into three main problems as discussed above: the *SAP*, the *OPRP*, and the *OBP*. Researchers have developed countless models and approaches to these problems, both integral and sequential, and there have been numerous case studies on the subject. This section elaborates on the most important works from literature on these topics.

2.1 Storage Allocation Problem

The SAP is defined by Reyes et al. (2019) as "the allocation of products into a storage space and optimisation of the material handling costs of storage space utilisation". This problem is not only applicable to e-grocery retailers, but to e-retailers in general. The obstacle for e-grocery FCs is that the number of storage locations is often higher, SKUs are much more volatile in terms of popularity, and there are many fragility and food safety guidelines compared to other e-retailers. The SAP is classified as an NP-hard problem, since the problem cannot be solved in polynomial time for large instances that closer resemble reality [Abdel-Hamid and Borndörfer (1994)]. Generally, there are three approaches used to solve the SAP throughout academic literature: (i) MILP models, (ii) heuristic models, and (iii) storage policies and rules.

Manzini et al. (2015) have created two MILP models, first for the assignment of SKUs to certain storage classes, and subsequently for assigning these classes to storage locations, based on demand patterns. They argue that these models can help make long-term decisions by considering the "life cycle" of the order picking process. Long-term decisions cannot be easily changed, and are decisions such as deciding the layout of the order pick circuit and what kind of system (automated or manual) to use. Others use a MILP model to solely focus on optimal SKU assignment in terms of travel time by the order pickers, as done by Ramtin and Pazour (2015). The approach that is used is assigning the most active SKUs (also called fast-movers) to the best pick positions for varying other parameters such as demand, differing peak hours and different system configurations. It can be seen from these examples that the application of the MILP model can vary and it is a versatile way to focus on many different objectives.

Apart from all heuristic measures, there are also meta-heuristics that are frequented throughout literature. Tabu search, genetic algorithms, and artificial neural networks are examples of those. Otto et al. (2017) uses tabu search to optimise the SAP in flow racks for ergonomics of the order pickers. Good ergonomics for order pickers entails taking into account if the order pickers need to bend and lift a lot and reducing this. Another case is the one from Yang et al. (2015), where they use a tabu search to reduce the complexity of the problem of location assignment in an automated storage/retrieval system, which shows strong results both in quality of the solution and the computational time needed.

Lastly, there are examples of research that makes use of storage policies and rules, such as *class-based strategy*, *correlation-based strategy* and *random strategy*. The work of Petersen et al. (2004) is an example of class-based strategy, where all SKUs are ranked according to their pick activity, and compared to a volume-based strategy. The pick activity of an SKU signifies the number of times the SKU is picked. The volume-based strategy is when SKUs are ranked according to their volume. They show that the time spent on fulfilment processes goes down with the new strategy, yet there is a trade-off between the number of classes and the performance. This strategy, which combines both volume and popularity of an SKU, is the Cube-per-Order Index (COI) strategy. SKUs are assigned into classes based on their COI value. An example of this for manual FCs is the work of Fontana et al. (2020).

For the assignment of MSL, there seems to be only one previous research that resembles the strategy of this research, being the one from Jiang et al. (2021). The focus is on the correlation between SKUs and minimising the distance between SKUs that have a high correlation. Correlation between SKUs conveys that SKUs are often ordered together. Promising results are shown, however, the instance size remains limited and it is not possible to multiply storage locations for SKUs at more than two locations. Also, the order picker routing in their research is unconstrained, whilst there are often constraints in place for order picker routing due to operational feasibility.

2.2 Order Picker Routing Problem

The OPRP is a variation of the well-known Travelling Salesman Problem (TSP). The OPRP is very often studied and there are researches dedicated to provide a literature review on this topic specifically [Masae et al. (2020a)]. The common denominator in these researches is the trade-off between optimality and operational feasibility. Truly optimal routes often are more complicated than heuristic routing policies. The order pickers have to be able to follow the set route easily, because having deviations from the predetermined tour will counteract the efforts made to reduce the total distance travelled.

The OPRP is dependent on the type of warehouse, i.e. the warehouse layout. This is determined by several aisle characteristics. The two biggest groups of warehouses are conventional warehouses, that are rectangular in shape with parallel picking aisles, and non-conventional warehouses, such as U-shaped or V-shaped layouts [Glock and Grosse (2012), Çelk and Süral (2014)]. To solve the OPRP there are again the different strategies of exact methods, (meta-)heuristic methods, and routing policies. In the work of Masae et al. (2020b), an exact method is compared to the S-shape routing policy for a conventional warehouse with and without a middle cross aisle. It was shown that the exact method had quite significant gains over the heuristic method of between 6% and 35% and that adding a middle cross aisle always leads to shorter routes. The argument is made that optimal order picker routing should always be preferred.

Most research on the OPRP is focused on static order picking, meaning that the order list is set and will stay the same throughout the order picking process. However, there also exist examples where dynamic order picking is examined, where the order list can be altered or updated during the order picking process. SKUs can be added to the order list. This is done by Lu et al. (2016), where they re-calculate the optimal route during the picking operations for any new order that arrives. The algorithm was tested via simulations and it showed that it can outperform certain static heuristics both when distance and time are used as proxies of performance.

2.3 Order Batching Problem

For e-retailers that sell small SKUs in terms of volume, the OBP is of importance. Because of this characteristic of the SKUs, one order picker collects multiple SKUs during one pick round, and even multiple orders. Often, order pickers are assisted in this via a rolling cart with separate bins to collect the SKUs for separate orders. For the sake of clarity these will all be referred to as pick carts. Which orders are batched together on one pick cart is naturally paramount to the efficiency of the order picking process.

With the OBP there are many different sub-problems that need to be handled, of which one of the most important is constructing the time windows for the order picking [Gil-Borrás et al. (2021)]. Orders are delivered to customers and customers expect the orders at a certain time. Orders that need to leave the FC around the same time, can be batched together, whilst others cannot. The time window should be as big as possible to allow for more flexibility in the OBP, yet adhering to delivery times is also a must [Gil-Borrás et al. (2020)]. The other problem that is most often considered is the proximity of the SKU locations to one another. If the SKUs of the orders that are batched together are in close proximity to each other, the order picker will have to travel less distance to fulfil all orders, which is desirable [Il-Choe and Sharp (2014)].

The objective of the OBP is often to reduce the time spent on order picking. Chen et al. (2018) show an approach to this objective through heuristics, which are independent of any warehouse parameters. The size of the batch, i.e. the number of orders on one pick cart, is of course of great influence on the service time per order, yet they show that with optimal batch sizes the heuristic outperforms traditional methods.

Other research emphasises the effect of solving the OBP for the distance travelled by order pickers. Öncan (2015) does this for a conventional warehouse through a MILP model and a tabu search algorithm, which they compared to empirical results. The comparison showed significant overlap in results, indicating that the proposed batching yields the decreased distance in real-life operations. They solved the OBP for three different routing policies: traversal, return, and midpoint, of which the traversal (entering an aisle and completely traversing it) yields the best results.

In case the OBP also takes the order pickers into account, other objectives could be to balancing the workload [Zhang et al. (2017)], minimising waiting times of the order pickers due to congestion [Hahn and Scholz (2017)], or minimising the turnover time for example [Tang and Chew (1997)].

3 Problem Statement

In this section, the fulfilment processes that influence the order picking process are described. There are multiple interconnected processes that affect the efficiency of the order picking process. Also, research objective is stated, as well as the problem setting with input, output, assumptions, and constraints. This research is done in collaboration with an European e-grocery retailer, which will be referred to as such. Their operational set-up is used as a guideline for this research.

3.1 Fulfilment Processes

In FCs, all processes are connected and one process influences another. In Figure 2 the different processes that influence order picking efficiency are shown and how they connect to one another. It also shows which processes are inside the scope of the research and which are outside the scope.
The *slotting service* determines both the storage location of the SKU as well as the dimensions of that storage location. This is a process that can take place daily, however not for every SKU. SKUs are changing storage locations based on a variety of reasons, such as congestion, seasonality, demand, and fragility.

The day before the order picking commences, the *tote service* takes care of two processes: allocating order totes (order crates) to orders and allocating SKUs to certain order totes, which is essentially a bin packing problem. The aim is to use as little order totes as possible. This objective determines which SKUs should go into which order tote. Optimising the tote allocation to ensure that SKUs that have a high correlation in characteristics such as storage location are placed together in an order tote is imperative to reducing the travelled distance of order pickers. The tote allocation that is used in the FCs for the case study already has this functionality and therefore it is outside of the scope of this research.

The *planning service* determines to which dispatch frame the order tote is assigned and the respective location of the order tote within the dispatch frame. This is dependent on the delivery routes and therefore a vehicle routing problem needs to be solved for this. The dispatch frame is the frame in which order totes are placed that is used for the home delivery service of the e-grocery retailer. This is part of the distribution process and is out of the scope for this research. An order can be comprised of several order totes, in case the customer has ordered a large number of SKUs. Since this research does not consider the tote service and order totes have the same composition throughout, one order tote is considered to be one order. Therefore, one order tote will be referred to as one order from here on out.

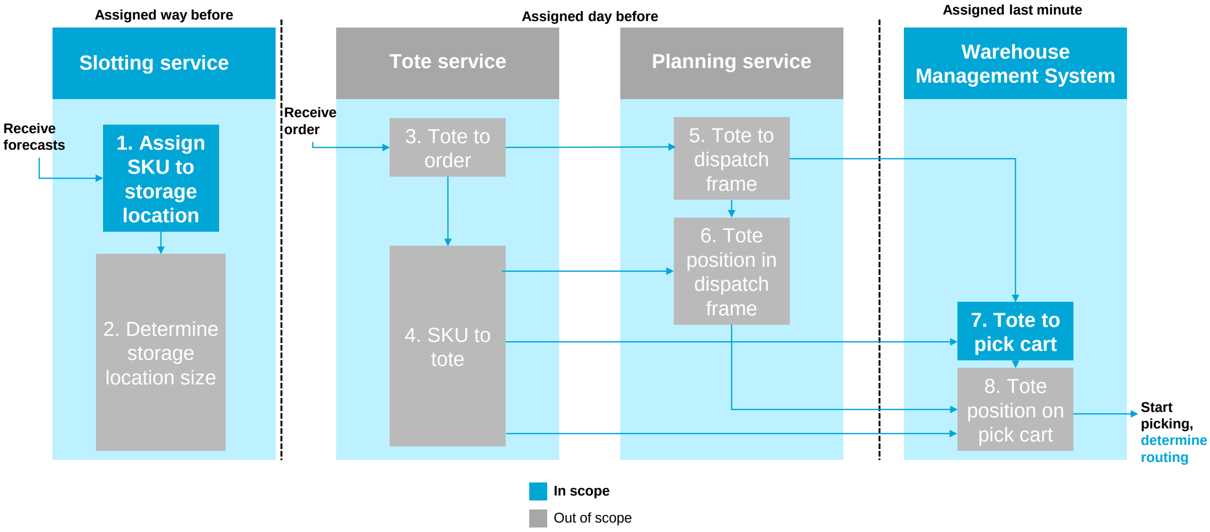


Figure 2: Interconnected processes that influence order picking efficiency.

Lastly, right before the pick round commences the *Warehouse Management System* (WMS) assigns the orders to pick carts, i.e. batches the orders together and also determines the location of the order on the pick cart, for both good ergonomics for the order pickers and efficiency. Afterwards, the pick round is ready to start and the routing of the order picker is set. Changes to the routing during the pick round due to discrepancies in WMS can take place in reality. However they should be treated case-by-case and are very dependent on a multitude of circumstances, therefore they are considered out of scope for this research.

3.2 Research Objective

The research objective is to reduce the distance travelled by order pickers throughout the pick circuit, to increase the efficiency of the order picking process. This is done by designing a tactical model that combines a set of orders, SKUs, and pick carts to determine the optimal FC operations. Therefore, it is determined which SKUs need to go into which zone of the order pick circuit. For this research, the specific place within the zone is of no importance. As mentioned, MSL per SKU are possible and its effects on reducing travelled distance are measured. Also, the routing of the order pickers is decided by establishing to which zone an order picker needs to go in order to fulfil its assigned orders. The routing within the zones is set by the *one-way S-shape* heuristic. There can be multiple entry- or exit points, dependent on SKUs that make up the orders. Moreover, it is determined which orders are the most beneficial to be batched together on a pick cart, such that similarities between orders are used to the advantage of the e-grocery retailer. This research is done in collaboration with a European e-grocery retailer which provides the necessary input data and certain operational constraints. The models are designed for conventional FCs, that have a number of multiple parallel aisles. There is no middle aisle to cross any other aisles. This layout and routing is shown in Figure 3.

3.3 Problem Setting

This subsection describes the full problem setting, what the output will look like and which assumptions are made and used throughout the models. Moreover, the operational constraints are discussed.

Input
The input data consists of three main sources:

- Order data: The order data comprises the orders, all different SKUs that are present in each order (i.e. the order lines per order, where an order line consists of the SKU that needs to be picked from the storage location and its quantity) and the truck that is used to transport the order from the regional FCs to the local hubs.
- SKU data: The SKU data contains all SKU characteristics, such as dimensions, demand, storage type, and current storage location. SKUs can be stored on five different types of locations, being locations for: (i) *pallets*, (ii) *shelving units*, (iii) *dollies*, (iv) *flowracks*, or (v) *roll containers*. It is dependent on the type of location how much space an SKU takes up in the pick circuit.
- Additional data: Some data on the layout of the different FCs, such as the number of aisles and length of the pick circuit.

Output
The output of the models is threefold:

- Zone allocation per SKU: The storage assignment problem is simplified to assigning SKUs to zones and not to specific storage locations in a zone.

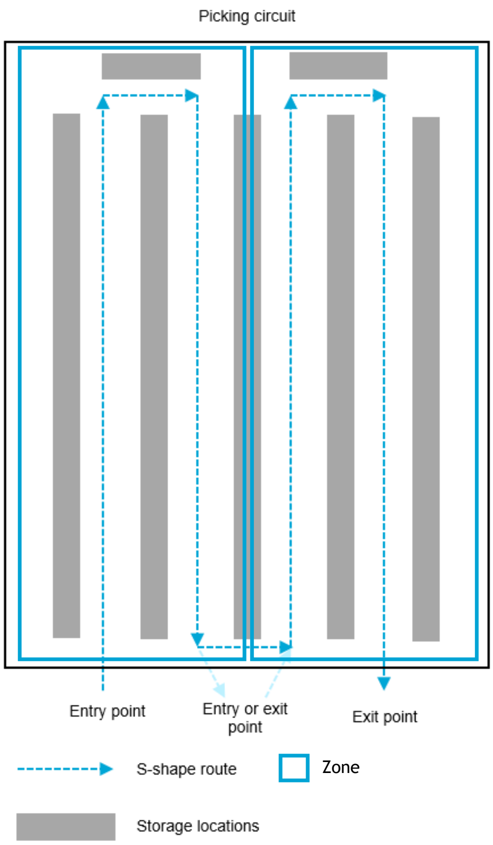


Figure 3: The generic layout of the conventional FCs with multiple parallel aisles, no middle aisle, multiple entry- and exit points, two zones, and one-way, S-shape routing.

A zone is comprised of several aisles in the FC. Whilst it is not done for this research, it is possible to optimise the storage allocation within a zone through the slotting service. This determines the exact dimensions of the storage location.

- Zones to travel to per order picker/pick cart: Each order picker takes one pick cart with them during their pick round. The solution of the routing problem will be which zones the order pickers travel to. The problem is defined with two zones, which will always be visited in a set sequential order. Within a zone the order picker will follow the one-way, S-shape routing strategy, as set by the European e-grocery retailer.

- Order allocation per pick cart: The batching problem is solved by grouping orders together on a pick cart. The orders on the pick cart need to traverse the same zone(s) and need to belong to the same truck shipment. The solution shows which orders are batched together on one pick cart.

Assumptions

Throughout the research, several assumptions are made. They are in line with the observations made by the FC of the European e-grocery retailer. First, it is assumed that there is always a pick cart available for an order picker. Since the number of pick carts is not limited, this can never constrain the solution. Moreover, the assumption is that there is no congestion in the order pick circuit despite multiple order pickers being in the circuit simultaneously. Because of this assumption it is possible to work with an average, empirical speed for order pickers, such that the time spent on order picking also can be calculated. With the time and the hourly labour costs the potential cost savings can be deduced. Next to this, it is assumed that there is always enough stock available on the storage location for the order picker to collect the SKU. This entails that there are never any routes needed that deviate from the set route determined at the start of the pick round. As mentioned, this ensures that detours are not needed and out of scope for the research. Also, it is assumed that an SKU takes up storage space in relation to its demand and volume. SKUs that have high demand or are bigger will need more storage space than an SKU that is hardly ever ordered or is small. How this relationship works exactly will be outlined in the mathematical model. There is no fragility constraint in place, meaning that it is not necessary to place SKUs that are fragile (such as eggs) in a zone previous to SKUs that are not fragile (such as a bottle of water). This is done because of the assumption that fragility constraints can be satisfied within a zone itself.

Operational Constraints

The model is subject to several constraints to ensure operational feasibility. They can be split into four

Table 1: Overview of sets in the MILP model.

Sets	
$\mathcal{I}$	Set of SKUs, indexed by i
$\mathcal{I}_o$	Subset of all SKUs i in order o
$\mathcal{Z}$	Set of zones, indexed by z
$\mathcal{O}$	Set of orders, indexed by o
$\mathcal{O}_{inc}$	Set of orders that are incompatible with one another with regards to batching them on pick carts
$\mathcal{P}$	Set of pick carts, indexed by p

Table 2: Overview of parameters in the MILP model.

Parameters	
L_z	The length of a zone [m]
C_z	The capacity of a zone [m^2]
D_i	The daily demand of an SKU [# picks/day]
FF_i	The front facing surface of an SKU [m^2]
MP_i	The storage type multiplication factor of an SKU [-]
SF_i	The storage type factor of an SKU [-]
PC_{max}	The maximum number of orders that can be placed on one pick cart [#]
MP_{max}	The maximum number of available SKU storage locations [#]

Table 3: Overview of decision variables in the MILP model.

Decision Variables	
m_p	Continuous, the number of walking meters per pick cart p
$t_{i,p}$	Binary, unitary if SKU i is on pick cart p
$u_{i,z}$	Binary, unitary if SKU i is placed in zone z
$v_{i,o,p}$	Binary, unitary if SKU i from order o is on pick cart p
$w_{o,p}$	Binary, unitary if order o is on pick cart p
$x_{p,z}$	Binary, unitary if pick cart p travels to zone z
$y_{i,p,z}$	Binary, unitary if SKU i from pick cart p is taken from zone z

main categories: SKU constraints, order constraints, zone constraints, and pick cart constraints. SKU constraints ensure that every SKU is assigned to at least one zone and that there is a maximum number of available places for all SKUs. Multiplying storage locations for one SKU has many of operational dependencies, so this can only be done for a selected number of SKUs. The order constraint needs to guarantee that each order gets assigned to a pick cart. An order cannot travel through the pick circuit "on its own", it needs to be an order on a pick cart. Next, the zone constraint handles the capacity of a zone. A zone cannot be filled with too many SKUs in such a manner that it will exceed its capacity. Therefore, the space needed for all SKUs assigned to the zone must be calculated and be lower than the capacity. Lastly, there are the pick cart constraints. These constraints make sure that the routing of the pick cart happens correctly and set the limit for the number of orders that can be batched together on one pick cart. Also, these constraints ensure that an SKU that has MSL is picked from one location only. Moreover, these constraints take care of the fact that all orders on a pick cart must belong to the same truck shipment. This is done in order to make sure that an order that is in the morning truck shipment cannot be paired with an order that is in the evening truck shipment. This is a loose way of applying a time constraint to the orders on the pick cart.

4 Methodology

The methods used for this research are elaborated in this section. First, the MILP model is discussed and how this model leads to an exact optimal solution to the problem. Next, the ALNS is described, along with its destroy- and repair mechanisms.

4.1 Mixed Integer Linear Programming Model

The exact model is formulated as a MILP model, meaning that the decision variables of the problem can be binary, integers, or continuous. The input data of SKU characteristics and historic order data will give an optimised SKU-to-zone, pick-cart-to-zone, and order-to-pick-cart assignment. The sets, parameters, and decision variables can be found in Table 1, Table 2 and Table 3, respectively. The mathematical formulation of the MILP model is shown next. The MILP formulation is solved by using a commercial solver. This solver makes use of the branch & bound technique, along with cutting planes, symmetry breaking and search-techniques and can therefore also be called a branch & cut technique. Branch & bound is a way of structuring feasible solution space, such that only part of the solutions need to be examined. Cutting planes are used to tighten the MILP relaxations [Lawler and Wood (1966)].

The objective function, Equation 1, minimises the sum of the walking meters traversed by each pick cart, i.e. the travelled distance of order pickers. This objective is subject to twenty constraints in total. Constraint (2) makes sure that each SKU is assigned to at least one zone, whilst it may be more than one zone. Constraint (3) ensures that the number of storage locations used for all SKUs across all zones is not above the threshold of maximum storage places available in the entire pick circuit. Constraint (4) guarantees that each order is assigned to a pick cart, whilst Constraint (5) warrants the fact that the capacity of a zone is not exceeded. This constraint is split into several parts and is dependent on the type of storage location. For example, if an SKU is stored at a pallet location, the amount of space it takes up is set and always the same. However, if an SKU is on a shelving unit, the amount of space it takes up is dependent on its volume and demand. This is handled by the storage parameters of the SKU in Constraint (5).

Constraint (6), (7), and (8) make sure that an SKU from a pick cart is only taken from a zone if that SKU is allocated to the zone, the SKU is allocated to the pick cart and the pick cart is allocated to the zone. Next, Constraint (9) establishes the maximum number of orders that can be batched together on one pick cart. Constraint (10) and (11) maintain that an SKU can only be on a pick cart if the SKU is in an order and that same order is on the pick cart.

To ensure that orders are only placed on a pick cart together if they belong to the same truck shipment, Constraint (12) is in place. This constraint makes use of an incompatibility list for all orders. Then, Constraint (13) ensures that if a pick cart is assigned orders it needs to traverse zones to pick up the needed SKUs. Furthermore, Constraint (14) gives the definition of the walking meters per pick cart, i.e. the travelled distance. If the pick cart travels to that zone, the length of that zone is added to the overall distance to calculate this. Lastly, Constraints (15) through (21) deal with the values that the decision variables can take.

$$\min \sum_{p \in \mathcal{P}} m_p \tag{1}$$

subject to

$$\sum_{z \in \mathcal{Z}} u_{i,z} \geq 1 \qquad \forall\, i \in \mathcal{I} \tag{2}$$

$$-MP_{max} + \sum_{z \in \mathcal{Z}, i \in \mathcal{I}} u_{i,z} \leq 0 \tag{3}$$

$$\sum_{p \in \mathcal{P}} w_{o,p} = 1 \qquad \forall\, o \in \mathcal{O} \tag{4}$$

$$-C_z + \sum_{i \in \mathcal{I}} (D_i \cdot FF_i \cdot MP_i + SF_i) \cdot u_{i,z} \leq 0 \qquad \forall\, i \in \mathcal{I} \tag{5}$$

$$-|\mathcal{I}| \cdot x_{p,z} + \sum_{i \in \mathcal{I}} y_{i,p,z} \leq 0 \qquad \forall\, p \in \mathcal{P}, z \in \mathcal{Z} \tag{6}$$

$$y_{i,p,z} - u_{i,z} \leq 0 \qquad \forall\, i \in \mathcal{I}, p \in \mathcal{P}, z \in \mathcal{Z} \tag{7}$$

$$-t_{i,p} + \sum_{z \in \mathcal{Z}} y_{i,p,z} = 0 \qquad \forall\, i \in \mathcal{I}, p \in \mathcal{P} \tag{8}$$

$$-PC_{max} + \sum_{o \in \mathcal{O}} w_{o,p} \leq 0 \qquad \forall\, p \in \mathcal{P} \tag{9}$$

$$v_{i,o,p} - w_{o,p} = 0 \qquad \forall\, o \in \mathcal{O}, i \in \mathcal{I}_o, p \in \mathcal{P} \tag{10}$$

$$-PC_{max} \cdot |\mathcal{I}| \cdot t_{i,p} + \sum_{o \in \mathcal{O}} v_{i,o,p} \leq 0 \qquad \forall\, i \in \mathcal{I}, p \in \mathcal{P} \tag{11}$$

$$w_{o_1,p} + w_{o_2,p} \leq 1 \qquad \forall (o_1, o_2) \in \mathcal{O}_{inc}, p \in \mathcal{P} \tag{12}$$

$$-PC_{max} \cdot \sum_{z \in \mathcal{Z}} x_{p,z} + \sum_{o \in \mathcal{O}} w_{o,p} \leq 0 \qquad \forall\, p \in \mathcal{P} \tag{13}$$

$$m_p - \sum_{z \in \mathcal{Z}} L_z \cdot x_{p,z} = 0 \qquad \forall\, p \in \mathcal{P} \tag{14}$$

$$m_p \geq 0 \qquad \forall\, p \in \mathcal{P} \tag{15}$$

$$t_{i,p} \in \{0,1\} \qquad \forall\, i \in \mathcal{I}, p \in \mathcal{P} \tag{16}$$

$$u_{i,z} \in \{0,1\} \qquad \forall\, i \in \mathcal{I}, z \in \mathcal{Z} \tag{17}$$

$$v_{i,o,p} \in \{0,1\} \qquad \forall\, i \in \mathcal{I}, o \in \mathcal{O}, p \in \mathcal{P} \tag{18}$$

$$w_{o,p} \in \{0,1\} \qquad \forall\, o \in \mathcal{O}, p \in \mathcal{P} \tag{19}$$

$$x_{p,z} \in \{0,1\} \qquad \forall\, p \in \mathcal{P}, z \in \mathcal{Z} \tag{20}$$

$$y_{i,p,z} \in \{0,1\} \qquad \forall\, i \in \mathcal{I}, p \in \mathcal{P}, z \in \mathcal{Z} \tag{21}$$

4.2 Adaptive Large Neighbourhood Search Model

Solving the MILP model for large data instances is not possible within acceptable time limits, due to the large size of the problem and the memory needed. Therefore, it is decided to use an alternative solution method as well, i.e. an ALNS method. The MILP model can provide a benchmark for the ALNS model, to show that the performance of the ALNS model is up to par for smaller instances and that the results for larger instances can be deemed trustworthy. First, the general ALNS model will be discussed and afterwards the destroy- and repair mechanisms are explained.

4.2.1 General ALNS

The ALNS model is an extension of the Large Neighbourhood Search (LNS) as designed by Shaw (1998) and was first introduced by Ropke and Pisinger (2006). The "adaptive" part refers to the use of different destroy- and repair mechanisms, instead of having only one type. The ALNS approach is chosen because of this specific characteristic, such that the heuristics that are chosen as destroy- and repair mechanisms are suitable for the problem. There are an infinite number of robust and fast heuristics to use and novel ones can also be incorporated, hence making the model very versatile. It is often hard to determine beforehand which heuristic will be most suitable to the problem. It might be the case that a heuristic is chosen, all instances are tested, and bad results are achieved. The ALNS combats the use of a bad heuristic by having multiple and adjusting the probabilities based on its past successes. If a heuristic is not successful in terms of the objective, the probability that it gets used lowers. The algorithm of the ALNS used in this research can be found in Algorithm 1.

Algorithm 1: Adaptive Large Neighbourhood Search

input: initial solution $s_{best} = s \in \{solutions\}$, $q \in \mathbb{N}$
while *stopping criteria is not met* **do**
    for i = *1,..., p_{update}* **do**
        select $r \in R$ and $d \in D$ according to probabilities p
        $s' = r(d(s))$ for q requests from s
        if *acceptation criterion is met* **then**
            $s = s'$
            if $f(s) < f(s_{best})$ **then**
                $s_{best} = s$
            end
        end
    end
    update weights w and probabilities p of the heuristics
end
return s_{best}

For this research, there are two initial solutions for the ALNS model. One is the current SKU-to-zone assignment of the European e-grocery retailer from which the data is obtained. This SKU-to-zone assignment already has been optimised, mainly for replenishment and fragility. Since these factors are not the subject of this research, it is interesting to see how they will compare. Moreover, the initial solution is of course already feasible.

The other initial solution is based on the correlation between SKUs. In this case, for each SKU combination it is determined if they are correlated (often ordered together) or not. Then, the SKUs with the highest correlation will be located together in one zone, whilst adhering to the capacity constraint of the zone. By comparing the results of the ALNS for these two different initial solutions, the impact of the initial solution on both computational time and performance can be measured.

A characteristic of the LNS is that a large neighbourhood is searched, as the name implies. Although this may increase the run time of the algorithm, it prevents the algorithm from getting stuck in local optima - to an extent. Another part of this is that also worsening solutions can be accepted and that the search is for solutions in the neighbourhood of s, instead of solely focusing on the neighbourhood of s_{best}. For the acceptance of solutions that are worse than the previous solution, a simulated annealing approach is used as designed by Van Laarhoven and Aarts (1987). The equation to calculate the probability that a worsening solution is accepted is seen in Equation 22.

$$p_{sa} = \exp^{-\left(\frac{c(s)-c(s')}{T}\right)} \tag{22}$$

To use this probability, the initial temperature must be set. This is done by solving the equation with the initial solution and a new solution that is w% worse, for an unknown temperature and setting the probability to 0.5. The 0.5 represents an acceptance probability of 50%. This entails that the first worsening solution that is exactly w% worse than the initial solution is accepted with a probability of 50%. w is the *start temperature control parameter*. The temperature is then cooled for every iteration by a factor c, the *cooling rate*, following a geometric decay.

As can be seen in Algorithm 1, destroy- and repair mechanisms are chosen according to their probabilities p. These probabilities and weights are updated every update period, which is set at 50 iterations. To calculate the probabilities, weights have been designed for the heuristics. The weights for each heuristic is determined by their previous successes. The weight of a heuristic can be decided by Equation 23, where w_{ij} is the weight of heuristic i that is used throughout update period j. The *reaction factor* r determines how strongly the weights are adjusted for their past successes. Next to that, π_i is the *success score* of the heuristic, which is calculated by summing σ_1, σ_2 and σ_3 of the heuristic. When these factors are assigned to the success score can be found in Table 4. The number of times the heuristic has been used over the last update period is depicted by θ_i.

$$w_{i,j+1} = w_{ij}(1-r) + r \cdot \frac{\pi_i}{\theta_i} \tag{23}$$

From the weights, the probabilities for each heuristic can be determined by using the *roulette wheel selection principle*. If there are n heuristics with each a

Table 4: Score factors.

Parameter	Description
σ_1	New solution is the new s_{best}
σ_2	New solution improves the current solution
σ_3	New solution does not improve the current solution, but is accepted

weight w_i, the probability that heuristic j is chosen is as depicted in Equation 24.

$$p_j = \frac{w_j}{\sum_{n}^{i=1} w_i} \tag{24}$$

It is decided to apply no noise to the objective function. Noise should randomise the insertion heuristics to discourage the same local moves. However, research shows that adding noise has little to no effect [Grimault et al. (2017)].

4.2.2 Destroy Mechanisms

The ALNS model uses three destroy mechanisms, or removal heuristics: (i) *random removal*, (ii) *worst removal*, and (iii) *fast-mover removal*. All three heuristics take a solution as an input, as well as the integer q. This integer depicts the number of requests that need to be removed and is determined by $q = \xi \cdot |R|$. Here, ξ represents a percentage, thus q is a percentage of all requests.

For random removal the procedure is quite straightforward: remove q requests and choose them randomly. It might not seem that this will quickly yield good solutions, however it provides diversification which is cardinal to exploring a very large search space.

The next destroy mechanism is called worst removal. For all requests it is calculated what the costs of the request is. This is determined by subtracting the cost of the solution *without* the request from the cost of the solution *with* the request. This is ordered in an array and through the parameter p_{worst} the removal of requests is randomised, to avoid the same requests being removed more often than others. The pseudocode of the algorithm can be found in Algorithm 2.

Algorithm 2: Worst Removal Heuristic

input: solution $s \in \{solutions\}$, $q \in \mathbb{N}, p \in \mathbb{R}$
while $q > 0$ **do**
  create array L: all requests sorted by descending $c(i, s)$
  create number y: random number in interval $[0, 1)$
  determine request $r : L[y^p \cdot |L|]$
  remove request r
  $q = q - 1$
end
return s_{des}

Lastly, there is fast-mover removal. This is a novel heuristic and is designed with e-grocery fulfilment operations in mind. This heuristic sorts the requests based on the number of times it is requested and then chooses a request of the top 60% at random. Again, this is to diversify the search. This heuristic is created with the thought in mind that requests that are often visited have a high impact on steering the objective. Making sure that these get removed and inserted at new locations will allow the algorithm to determine their best locations. The algorithm is seen in Algorithm 3.

Algorithm 3: Fast-Mover Removal Heuristic

input: solution $s \in \{solutions\}$, $q \in \mathbb{N}$
create array F: all requests sorted by visit frequency
while $q > 0$ **do**
  create number y: random number in interval $[0, 0.6 \cdot |F|)$
  determine request $r : F[y]$
  remove request r
  $q = q - 1$
end
return s_{des}

The heuristic gets it name fast-mover removal from the fact that certain requests, or SKUs, are so-called fast-movers: SKUs that are often ordered and therefore "move fast" through the FC. The choice is made to use this characteristic of an SKU due to the Pareto principle that also applies to SKUs. Analysis of the orders showed that the 20% most ordered SKUs are found in around 80% of the orders. Therefore, moving them will have a big influence on the objective.

4.2.3 Repair Mechanisms

Next to destroy mechanisms there are repair mechanisms. The repair mechanisms used in this research are all *parallel* insertion heuristics, meaning that routes are build at the same time, without having to recalculate for each insertion. Again, three heuristics have been used: (i) *random insertion*, (ii) *cheapest insertion*, and (iii) *swap insertion*.

Random insertion is very similar to random removal. The removed request is taken and inserted into the solution at a new, random location.

Cheapest insertion is very similar to the destroy heuristic of worst removal. The algorithm decides what the costs are to insert a request at its best position, by comparing it to the costs without the request. Then, all requests are inserted at their minimum cost position and the new solution is returned. The pseudocode is seen in Algorithm 4.

Algorithm 4: Cheapest Insertion Heuristic

input: solution $s \in \{solutions\}$, $q \in \mathbb{N}$, $i_r \in I$
create array I: all removed requests i, sorted by descending $c(i, s)$ at their best new location
while $q > 0$ **do**
  determine request $r : I[q]$
  insert request r at best location
  $q = q - 1$
end
check feasibility
return s'

Swap insertion is also a relatively simple insertion heuristic. It swaps the locations of removed requests. For example, if removed request a was at location x and removed request b was at location y, the repaired solution consists of request a at location y and request b at location x. The algorithm can be found in Algorithm 5.

Algorithm 5: Swap Insertion Heuristic

input: solution $s \in \{solutions\}$, $q \in \mathbb{N}$, $i_r \in I$
for *removed requests* **do**
  if *previous location of request* $= x$ **then**
    insert request at location $x + 1$
  end
  else if *previous location* $= |x|$ **then**
    insert request at location 0
  end
end
check feasibility
return s'

For each algorithm it holds that the feasibility of the new solution needs to be checked. This is done by examining the total number of storage locations used and the zone capacities. The truck constraint is already adhered to by the way the cost function is constructed. If a solution is not feasible, it gets (randomly) deconstructed and rebuild until it is.

5 Computational Experiments

This section describes the computational experiments. First, the test instances and their differences are laid out. Next, the parameter tuning is discussed and the final parameters used for the test instances are described. For the computational experiments the models are implemented in Python and the MILP model is solved through the commercial solver Gurobi, version 9.5.1, in combination with Python version 3.10. The models are run on a Dell Latitude 5411, with 32 GB of RAM and an Intel(R) Core(TM) i7 processor. Running the experiments on this laptop did not always yield even a feasible solution for larger instances within the specified computational time. Therefore, a server was used to increase computational power. The server is a dual AMD EPYC 7551 server with 64 cores, 128 threads and 256 GB of RAM and was made available by Delft University of Technology. The time limit that is set for all experiments is 7200 seconds, or two hours.

5.1 Test Instances

Several instances were used to test the performance of the models. The instances are chosen due to the diversity on three fronts: diversity in terms of instance size, i.e. number of orders, diversity in FCs, and diversity in the type of day. The first front will affect the computational time needed for the models to solve the instance. From this it can be gathered how the models are affected by differing instance sizes. The next two fronts should not lead to differing results and to test this hypothesis the different instances are used. Especially the type of day is important to watch, since routing, order batching, and SKU storage assignment are things that cannot be changed daily. Hence, ensuring that these are near optimal regardless of the day of the week, is cardinal. Differences between FCs may happen, however, standardised practices across FCs are preferred from an operational point of view. The instances and their characteristics can be found in Table 5. For all instances the number of MSL is set to 50, in line with operational feasibility.

The data is provided by the European e-grocery retailer and the characteristics of two of their actual FCs have been used as input parameters. The FCs have been named and ordered randomly.

Table 5: Names, definitions and characteristics of the different test instances.

Instance	FC	# of orders	Day	# of ols
fcA_50_s	FC A	50	Sunday	683
fcB_50_s	FC B	50	Sunday	512
fcA_100_m	FC A	100	Monday	1035
fcB_100_m	FC B	100	Monday	1149
fcA_400_w	FC A	400	Wednesday	4106
fcB_400_w	FC B	400	Wednesday	4162
fcA_600_t	FC A	600	Thursday	6666
fcA_600_f	FC A	600	Friday	6635
fcA_3000_s	FC A	3000	Sunday	32493
fcB_3000_s	FC B	3000	Sunday	34230

5.2 Parameter Tuning

Before results of the models can be shown and analysed, the correct parameters need to be set for the ALNS model. An overview of all parameters is seen in Table 6.

Table 6: Parameters and their definition.

Parameter	Definition
p_{update}	Update period for weights and probabilities
ξ	Percentage of the neighbourhood that is destroyed and repaired
w	Start temperature control parameter for the simulated annealing
c	Cooling rate of the temperature for the simulated annealing
σ_1	Success of a heuristic for new best found solution
σ_2	Success of a heuristic for improving solution
σ_3	Success of a heuristic for worsening solution but accepted
r	Reaction factor that controls the adjustment of weights to changes
p_{worst}	Parameter for randomisation of worst removal heuristic

The choice has been made to only tune the parameters of ξ and w for small instances and to choose the values used in Ropke and Pisinger (2006) for the remaining parameters. This is done to limit the time spent on parameter tuning and to focus on the parameters that could influence the performance the most as observed during testing. For the two parameters that will be tuned the same approach is used as the approach used by Žulj et al. (2018). A good parameter setting that was identified during testing is set as the initial parameter setting. Then, a single parameter is changed, whilst the other remains fixed. Five runs are performed for the test instances and the setting with the best average result is chosen. Then, the same is repeated for the next parameter with the previously determined parameter fixed at its best setting.

First, the parameter tuning is done for ξ, whilst fixing w at 0.05. The parameter ξ is changed between five values and the results are shown in Table 7.

Table 7: Varying ξ for the test instances, results are the rounded averages of five runs. Best result per instance is in bold. w is set at 0.05.

Instance	0.05	0.1	0.2	0.3	0.4
fcA_50_s	878	805	829	**793**	**793**
fcB_50_s	798	779	**760**	**760**	**760**
fcA_100_m	1537	1624	**1476**	1512	1512
fcB_100_m	1368	**1254**	**1254**	1292	**1254**
fcA_400_w	6722	6697	6636	**6612**	**6612**
fcB_400_w	4142	4180	**4104**	**4104**	4142

Table 8: Varying w for the test instances, results are the rounded averages of five runs. Best result per instance is in bold. ξ is set at 0.3.

Instance	0	0.01	0.025	0.05
fcA_50_s	805	829	**793**	**793**
fcB_50_s	798	**760**	798	**760**
fcA_100_m	**1475**	1487	1559	1512
fcB_100_m	**1216**	**1216**	1277	1292
fcA_400_w	6649	6636	**6600**	6612
fcB_400_w	4123	4142	**4104**	**4104**

The trend for ξ is not very pronounced. Both the values of 0.2 and 0.3 seem good contenders. Since some instances have a large number of SKUs, it is decided to choose 0.3 for ξ, to keep the neighbourhood sufficiently large. This value is set and the tuning for w is continued in Table 8. Recall that w is the start temperature control parameter. If this is set at 0.05 it entails that an early solution that is 5% worse than the initial solution in terms of the objective function will be approved with a probability of 50%. Changing this value can change the time that is spent on worsening solutions. For w four different values have been tested, one of which is 0. Setting w to zero effectively means that the simulated annealing approach is not used and that solutions are only accepted if they have a positive effect on the objective. From Table 8 it can be seen that the trend for w is even less pronounced than for the previous parameter. However, on average the results are better for higher values of w. Of the higher values 0.05 is chosen, because of how the objective is determined. Since the objective is dependent on entire zones of the pick circuit being skipped (or not), a solution that is very close to the current best solution can have quite a higher objective value. Because of this reason it makes more sense to adopt a less strict simulated annealing acceptance. With this, all parameters have been set and the results can be found in Table 9. With this parameter setting for the ALNS model all test instances are be run.

Table 9: Parameter values used for the ALNS model.

Parameter	Value
p_{update}	50
ξ	0.3
w	0.05
c	0.99975
σ_1	33
σ_2	9
σ_3	13
r	0.1
p_{worst}	3

6 Results

This section will show the results that were obtained by running the computational test instances from the previous section. First, the results of the MILP model and ALNS model with different initial solutions are depicted. Afterwards, the managerial insights that are gathered from these results are discussed.

6.1 Computational Results

For all different instances, the MILP model is compared to the ALNS model, that has the current SKU-to-zone set up from the European e-grocery retailer as initial solution. The results can be seen in Table 10.

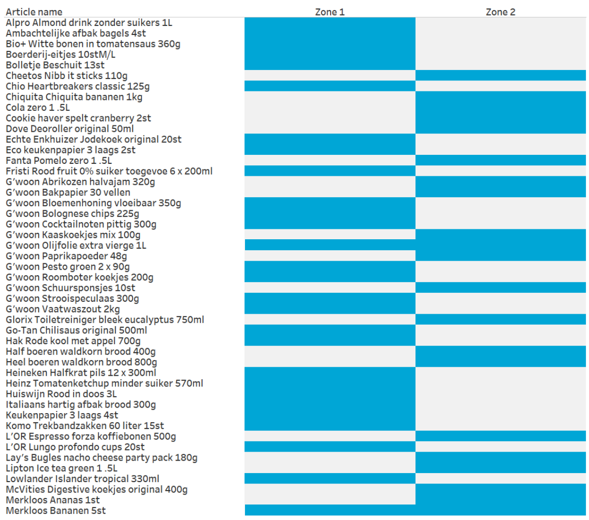


Figure 4: Results that show to which zone an SKU is assigned. In the example two SKUs have MSL, being olive oil and bananas.

Apart from the computational results, it is interesting to see the results of the SKU-to-zone assignment. Part of these results are seen in Figure 4, with an example of an SKU with MSL. From the results in Table 10 it can be seen that the MILP is computationally efficient for small instances, yet can take what

Table 10: Results of the computational experiments for the MILP model and ALNS model with current SKU-to-zone assignment as initial solution. Best incumbent (B.I.), lower bound (L.B.), optimality gap (O.G.) and time to best (T.T.B) are shown for the MILP model. For the ALNS model the best found solution (B.F.S), the computational time (C.T.), the rounded average (Avg), and standard deviation (S.D.) of five different runs are given. Not applicable (N.A.) is when no solution was obtained. *: instance run on the server of the Delft University of Technology for the MILP model.

Instance	**MILP**				**ALNS**			
	B.I. [m]	L.B. [m]	T.T.B. [s]	O.G. [%]	B.F.S. [m]	C.T. [s]	Avg [m]	S.D. [m]
fcA_50_s	610	610	50	0	793	164	793	0
fcB_50_s	570	570	17	0	760	158	760	0
fcA_100_m	1220	1220	343	0	1464	390	1512	80
fcB_100_m	950	950	1203	0	1140	459	1292	159
fcA_400_w	8296	3485	324	58	6588	877	6612	33
fcB_400_w	4180	3800	6153	9	3990	926	4104	170
fcA_600_t*	11163	5307	6181	52	9882	1463	10089	153
fcA_600_f*	10919	5307	6823	51	9455	2137	9686	175
fcA_3000_s*	N.A.				50081	6459	50386	342
fcB_3000_s*	N.A.				46930	5811	47082	159

is considered to be a long time to optimise for large instances. This occurs because of the solution space grows in size for large instances, i.e. there are more solutions to investigate before the optimal one can be chosen. Because of the computational complexity, the optimal solution was not found for all instances within the set time limit. The ALNS model is not necessarily quicker than the MILP model when it comes to small instances, however it is very close to the optimal objective. The performance regarding the objective value of the ALNS is good. The computational time advantage does come into play for bigger instances. For bigger instances (>100 orders) the ALNS takes, on average, 14.8% less time to reach 12.5% better solutions. For bigger instances the ALNS does become a bit more volatile. This can be explained by the fact that there are much more options for the model and the randomness of the algorithm has more influence.

How the objective value changes over time for the MILP can be seen in Figure 5 for two small instances. It is seen that the model converges quite quickly, yet that there are many big "steps" in lowering the objective. This can be explained by the fact that the objective value is based on the travelled distance per pick cart and a pick cart needs to traverse at least one zone. Adding or removing one pick cart entails that at least the entire length of one zone is added to or removed from the objective function.

A comparison also can be made between the ALNS models with different initial solutions. As mentioned, there are two types of initial solutions for the ALNS model: the SKU-to-zone assignment as currently used by the European e-grocery retailer or the SKU-to-zone assignment based on the correlation between SKUs. The computational results for the ALNS with the latter initial solution can be seen in Table 11. It is seen that the objective values are not necessarily better, yet, the results are obtained quicker.

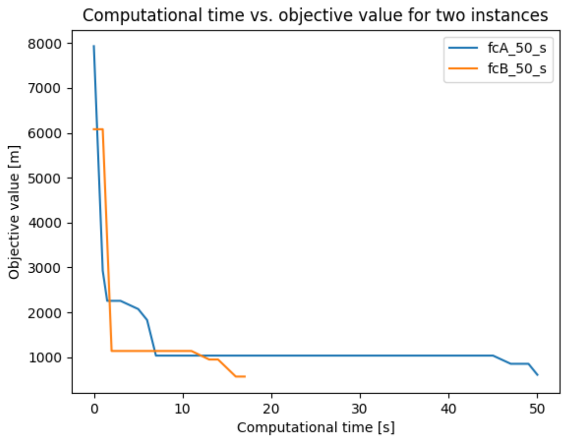


Figure 5: Objective value over time for the MILP for fcA_50_s and fcB_50_s.

Table 11: Results of the computational experiments for the ALNS model with correlated SKU-to-zone assignment as initial solution. This table can be compared to the right hand side of Table 10.

Instance	**B.F.S. [m]**	**C.T. [s]**
fcA_50_s	610	136
fcB_50_s	760	119
fcA_100_m	1464	293
fcB_100_m	950	407
fcA_400_w	6588	623
fcB_400_w	3990	1020
fcA_600_t	9943	1198
fcA_600_f	9455	1843
fcA_3000_s	50081	4287
fcB_3000_s	47120	5311

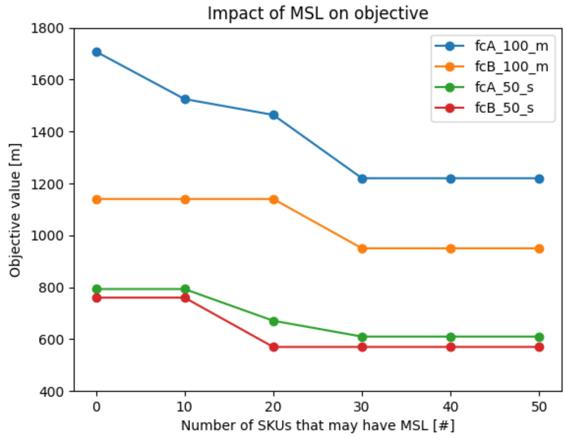


Figure 6: Sensitivity analysis of the number of MSL on the travelled distance.

6.2 Managerial Insights

From a managerial perspective, the results are interesting regarding the SKUs that have MSL. The outcome of this is that the SKUs that have MSL in the best found solutions are generally fast-mover SKUs. This result was expected since they make up a significant portion of the orders. It is also shown that the models are applicable to different FCs, by changing a few input parameters. This entails that e-grocery retailers that have multiple FCs can use the models across all FCs and still maintain robust results.

To make a business case from these results they can be compared to the actual travelled distance by order pickers of the European e-grocery retailer for the same set of orders. This comparison yields an average improvement of 18%. These distance savings, together with an average pick speed of order pickers and their hourly labour costs, can be converted to cost savings. However, data for average pick speed and hourly costs of an order picker is confidential and the actual cost savings are therefore not shared.

When looking at this research from an operational point of view, the models show that there is still a lot of unexplored potential left regarding order picking, especially when using MSL. However, order picking is not the only process that is happening throughout fulfilment operations. It might be the case that gains that are yielded in the order picking part are nullified or severely lowered by disadvantages upstream of order picking in the supply chain. These effects should be quantified to draw solid conclusions on whether to incorporate MSL or not. One of the biggest upstream disadvantages that can take place is in the receiving and replenishment process. The receiving operation consists of accepting the goods of the suppliers at the docks and scanning them in order to move on to the replenishment, the restoration of stock to a threshold level. However, if there are MSL for the SKU, the decision needs to be made on how to split the stock that comes into the FC. It should be known which percentage of the stock needs to go where and how the stock splitting will go, if it is at the dock or during the replenishment rounds. For these problems, the choice is between developing heuristics or using an exact model based on the forecasted demand of the SKUs.

A small sensitivity analysis has been executed to study the effect of the number of MSL on the reduction of travelled distance. For this, the number of MSL have been varied from 0 to 50. The results are shown in Figure 6.

It is seen increasing the number of SKUs that may have MSL has a positive effect on the reduction of travelled distance of order pickers. There are, however, some "tipping point". These tipping points show that further reduction of walking meters is only obtained when the possibility for MSL is above that tipping point. For operations it is important to know where the tipping point is, such that good decisions can be made on how many MSL to introduce. An FC does not want to add more MSL when operational complexity is increased because of this and no (further) reduction of travelled distance is realised.

7 Conclusions

This paper presented an integral approach to the SAP, OPRP, and OBP, whilst applying MSL for a select number of SKUs. An exact MILP model was created that could find a solution for instances up to 600 orders within acceptable time. Next to that, an ALNS model was created to find solutions for larger instances that are frequently occurring at e-grocery FCs in significantly less computational time. The models were tested on the problems for different instances and achieved good results in an appropriate time.

When comparing the models, it can be seen that the ALNS outperforms the MILP model in terms of time, whilst finding near-optimal solutions. The results show that travelled distance can be reduced by introducing MSL and taking an integral approach to order picking problems by, on average, 18%, with regards to current operations of an European e-grocery retailer. This ultimately can lead to an increase in efficiency, which could serve the e-grocery retailers that want to offer more supply but are constrained by their order picking capacity. The proposed SKUs that will have MSL are the fast-moving SKUs, showing that their placing and characteristics have such an impact on order picking efficiency that they cannot be underestimated.

Regarding the models, methodological or algorithmic improvements to both are possible. For the MILP model, there is the case of symmetry. Because the objective value is dependent on the routes of the pick carts, that can have more than 150 SKUs, changing the storage location of one SKU has little to no effect on the entire route of the pick cart. Therefore, many solutions exist where SKUs are reshuffled without affecting the objective value. For ALNS models, it holds that the increasing the neighbourhood size can help with this. For the MILP model, it could be beneficial to introduce symmetry-braking constraints, for example defining index precedence constraints among different SKU locations to limit equivalent combinations. Furthermore, the performance in terms of computational time of the ALNS is dependent on the initial solution. The initial solution where correlated SKUs are placed together in a zone is preferred over the initial solution where the current SKU-to-zone assignment of the European e-grocery retailer is taken, that has been optimised for replenishment and fragility. Ensuring that an initial solution is well-suited to the problem is therefore of importance to the performance of the ALNS.

Whilst these results show promise in regards to order picking, it must be acknowledged that there may be disadvantages to other parts of the fulfilment processes because of this. Replenishment of the SKUs that have MSL is much more difficult than SKUs that have only one storage location. Adjustment of the WMS should be considered for this. Also, because of several assumptions of the model such that there is always available stock, the reduction of travelled distance may be higher in theory than they would be in reality. It is important to test the models in practice at an e-grocery retailer that has similar characteristics as the one studied in this research, and retrieve empirical results that can be cross-referenced with the theoretical ones. In this way, the true impact of the propositions of the model can be measured.

For future research it would be very interesting to assess the impact of MSL on congestion of the order pickers. The hypothesis is that MSL will split the demand of an SKU over multiple locations and therefore less order pickers need to be at the same spot at the same time to collect that specific SKU. This warrants less congestion at certain locations. Although this hypothesis cannot be validated with the current models, it is very important to investigate when looking at the productivity of an FC. Further interests lies in seeing if adjusting the bin packing problem of SKUs to orders would increase the reduction of travelled distance. Since the models worked with a fixed set of orders, adjusting the order content (i.e. which SKUs are in the order) to make each order more suitable for the determined SKU-to-zone allocation could yield even better results. Moreover, if there would be more freedom in terms of splitting orders over multiple pick carts, further potential might be gained. The problem size of the OBP does grow because of this, however one can use the similarities between (parts of) orders to their advantage. Furthermore, developing a matheuristic combination of the MILP and ALNS model would be of interest. A matheuristic approach is one that has both an exact part and a (meta-)heuristic part. If the ALNS would be performed first up to a certain, acceptable, time limit, this could be used as an input for the MILP model. This could potentially reduce the computational time needed for the MILP model, whilst an optimal solution is reached. Lastly, seeing how the length of each zone influences the overall performance can be of interest. If zone length was dependent on the number of SKUs that were assigned to it, the models would become much more dynamic and this would give more managerial insights into how the FC layout in terms of zones should look like.

References

Abdel-Hamid, A. A.-A. and Borndörfer, R. (1994). On the complexity of storage assignment problems.

Alfonso, V., Boar, C., Frost, J., Gambacorta, L., and Liu, J. (2021). E-commerce in the pandemic and beyond. *BIS Bulletin*, 36(9).

Çelk, M. and Süral, H. (2014). Order picking under random and turnover-based storage policies in fishbone aisle warehouses. *IIE transactions*, 46(3):283–300.

Chen, F., Wei, Y., and Wang, H. (2018). A heuristic based batching and assigning method for online customer orders. *Flexible Services and Manufacturing Journal*, 30(4):640–685.

Dukic, G. and Oluic, C. (2007). Order-picking methods: improving order-picking efficiency. *International Journal of Logistics Systems and Management*, 3(4):451–460.

Fontana, M. E., López, J. C. L., Cavalcante, C. A. V., and Noriega, J. J. S. (2020). Multi-criteria assignment model to solve the storage location assignment problem. *Revista Investigation Operational*, 41(7):1019–1029.

Füchtenhans, M., Grosse, E. H., and Glock, C. H. (2021). Smart lighting systems: state-of-the-art and potential applications in warehouse order picking. *International Journal of Production Research*, 59(12):3817–3839.

Gil-Borrás, S., Pardo, E. G., Alonso-Ayuso, A., and Duarte, A. (2020). Grasp with variable neighborhood descent for the online order batching problem. *Journal of Global Optimization*, 78(2):295–325.

Gil-Borrás, S., Pardo, E. G., Alonso-Ayuso, A., and Duarte, A. (2021). A heuristic approach for the online order batching problem with multiple pickers. *Computers Industrial Engineering*, 160:107517.

Glock, C. H. and Grosse, E. H. (2012). Storage policies and order picking strategies in u-shaped order-picking systems with a movable base. *International Journal of Production Research*, 50(16):4344–4357.

Grimault, A., Bostel, N., and Lehuédé, F. (2017). An adaptive large neighborhood search for the full truckload pickup and delivery problem with resource synchronization. *Computers & Operations Research*, 88:1–14.

Hahn, S. and Scholz, A. (2017). Order picking in narrow-aisle warehouses: A fast approach to minimize waiting times. *Working Paper Series*.

Il-Choe, K. and Sharp, G. (2014). Small parts order picking: design and operation.

Jiang, W., Liu, J., Dong, Y., and Wang, L. (2021). Assignment of duplicate storage locations in distribution centres to minimise walking distance in order picking. *International Journal of Production Research*, 59(15):4457–4471.

Lawler, E. L. and Wood, D. E. (1966). Branch-and-bound methods: A survey. *Operations research*, 14(4):699–719.

Lu, W., McFarlane, D., Giannikas, V., and Zhang, Q. (2016). An algorithm for dynamic order-picking in warehouse operations. *European Journal of Operational Research*, 248(1):107–122.

Manzini, R., Accorsi, R., Gamberi, M., and Penazzi, S. (2015). Modeling class-based storage assignment over life cycle picking patterns. *International Journal of Production Economics*, 170:790–800.

Masae, M., Glock, C. H., and Grosse, E. H. (2020a). Order picker routing in warehouses: A systematic literature review. *International Journal of Production Economics*, 224:107564.

Masae, M., Glock, C. H., and Vichitkunakorn, P. (2020b). Optimal order picker routing in a conventional warehouse with two blocks and arbitrary starting and ending points of a tour. *International Journal of Production Research*, 58(17):5337–5358.

Öncan, T. (2015). Milp formulations and an iterated local search algorithm with tabu thresholding for the order batching problem. *European Journal of Operational Research*, 243(1):142–155.

Otto, A., Boysen, N., Scholl, A., and Walter, R. (2017). Ergonomic workplace design in the fast pick area. *OR Spectrum*, 39(4):945–975.

Petersen, C. G., Aase, G. R., and Heiser, D. R. (2004). Improving order-picking performance through the implementation of class-based storage. *International Journal of Physical Distribution & Logistics Management*.

Ramtin, F. and Pazour, J. A. (2015). Product allocation problem for an as/rs with multiple in-the-aisle pick positions. *IIE Transactions*, 47(12):1379–1396.

Reyes, J., Solano-Charris, E., and Montoya-Torres, J. (2019). The storage location assignment problem: A literature review. *International Journal of Industrial Engineering Computations*, 10(2):199–224.

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

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In *International conference on principles and practice of constraint programming*, pages 417–431. Springer.

Tang, L. C. and Chew, E.-P. (1997). Order picking systems: batching and storage assignment strategies. *Computers & Industrial Engineering*, 33(3-4):817–820.

Van Laarhoven, P. J. and Aarts, E. H. (1987). Simulated annealing. In *Simulated annealing: Theory and applications*, pages 7–15. Springer.

Yang, P., Miao, L., Xue, Z., and Qin, L. (2015). An integrated optimization of location assignment and storage/retrieval scheduling in multi-shuttle automated storage/retrieval systems. *Journal of Intelligent Manufacturing*, 26(6):1145–1159.

Zhang, J., Wang, X., Chan, F. T., and Ruan, J. (2017). On-line order batching and sequencing problem with multiple pickers: A hybrid rule-based algorithm. *Applied Mathematical Modelling*, 45:271–284.

Zhang, R.-Q., Wang, M., and Pan, X. (2019). New model of the storage location assignment problem considering demand correlation pattern. *Computers & Industrial Engineering*, 129:210–219.

Žulj, I., Kramer, S., and Schneider, M. (2018). A hybrid of adaptive large neighborhood search and tabu search for the order-batching problem. *European Journal of Operational Research*, 264(2):653–664.

II

Literature Study
previously graded under AE4020

Literature Research

Researching Multiple Storage Locations in an E-grocery Fulfilment Centre

AE4020: Literature Study

Merel Sterk

Delft University of Technology

TU Delft

Literature Research
Researching Multiple Storage Locations in an E-grocery Fulfilment Centre

by

Merel Sterk

This literature study is submitted in fulfilment of the course AE4020

Supervisor:	A. (Alessandro) Bombelli
Institution:	Delft University of Technology
Place:	Faculty of Aerospace Engineering, Delft
Project Duration:	January, 2022 - April, 2022
Date:	Friday 15th of April, 2022

Cover Image: Library, *Unsplash 2022*

TU Delft

Contents

Nomenclature

Abbreviations

Abbreviation	Definition
B2B	Business To Business
B2C	Business To Consumer
BP	Binary Programming
COI	Cube-Per-Order Index
DC	Distribution Centre
EPV	Electric Picnic Vehicle
FC	Fulfilment Centre
GAP	Generalised Assignment Problem
LMDP	Last-Mile Delivery Problem
MILP	Mixed Integer Linear Programming
MPP	Master Planning Process
NLP	Non-Linear Programming
OBP	Order Batching Problem
OPR	Order Picker Routing
PP	Picking Productivity
RC	Roll Container
RMP	Reduced Master Problem
SAP	Storage Assignment Problem
SA&TDE	Storage Assignment & Travel Distance Estimation
SKU	Stock Keeping Unit
SPI	Seconds Per Item
TSP	Travelling Salesman Problem
UPH	Units Per Hour
VRP	Vehicle Routing Problem
WMS	Warehouse Management System

List of Figures

List of Tables

1 Introduction

The rise of the online markets has grown extensively since the start of the new millennium, together with the rise of the internet and the availability of personal computers. Nowadays, it is possible to order almost everything and anything online, from electronics to furniture to groceries. Currently, the market of online groceries, or e-groceries, is still relatively small. Only around 8% of all the grocery shopping in The Netherlands is done online [1]. Of the supermarkets that offer online services, newcomer Picnic has a market share of around 20% [2]. The competition between supermarkets for their online market share is fierce, with also big retailers like Albert Heijn and Jumbo competing. The reason for a fierce competition is partly due to the big increase of the market as well due to the continuous growth. In 2020 the market grew with 65% in comparison to the previous year, to a total of 2.5 billion euros in The Netherlands alone [2]. With this growth in online ordering, there is also a rise in the number of fulfilment centres (FCs) or warehouses that need to accommodate for the offered services. FCs are different from warehouses, of which the main function solely is storing products. FCs handle all the stages of the order fulfilment, such as, but not limited to, storage, order picking and dispatching [3]. These operational processes in the FCs require time as well as funds. Therefore, often the main objective for management is optimising the FC processes, to decrease both of these objectives simultaneously.

Designing an FC and its layout is considered to be a very complex task, that can be very specific for each different FC [4]. This becomes more difficult as well as more important when the number of products in the FC grows. There are many possibilities and strategies on where to place which product and how much space in the FC is dedicated to certain operational activities. When looking at the case of an e-grocery FC, the number of products are usually very high, thus making it a complex problem. Out of all the operational processes taking place in the FC, order picking is the most labour-intensive operation. Order picking is the process of retrieving products from storage in response to a specific customer request [5]. Routing the order pickers from product to product is a form of the travelling salesman problem, which has been heavily studied [6]. The retrieval of products can either be done manually or automatically, with people or through machines. Another part of the design of a FC is the storage assignment, or the product placement within the FC. While this is not an operational process itself, the efficiency of the operational process of order picking is dependent on it. The combination of product placement within the FC (storage assignment) and the routing from product to product (order picking) are intertwined and an important factor for the operational efficiency.

Because of the big impact that both these problems, the storage assignment problem and the order-picker routing problem, have, improving them can have significant impact on the performance and productivity of an FC. In many FCs, the storage assignment is based on certain storage policies and rules, without relying on a specific algorithm [7]. Especially, having duplicate, or multiple, storage locations for the same product has not been implemented or researched. Therefore, the objective of the research is:

Improving the storage location assignment by using multiple storage locations in an e-grocery fulfilment centre, such that the routing of order pickers can be done in the most efficient way in terms of time.

The first model that will be developed will be an exact model. This entails that the model will search for the absolute best solution. There are different kind of exact formulations to develop a model, such as mixed integer linear programming (MILP), binary programming (BP) and non-linear programming (NLP) formulations. The solution techniques that can be used for these formulations are for instance column generation or branch & bound algorithms [7]. In this research the focus is on a MILP formulation, where the objective and constraints are all linear equations. However, some variables are integers, whilst others are non-integers, hence the mixed integer name. The different solution techniques that exist for a MILP model will be discussed and compared, to decide which technique is the most effective in solving the model. Since the problem size is very large, meta-heuristics are also considered. Meta-heuristics do not necessarily give the absolute best solution, yet can come very close to the optimal solution within a limited amount of time because of some simplifications [8]. Since there are many types of meta-heuristics with different advantages and disadvantages, these are analysed and discussed to chose the best approach for the problem at hand.

The goal of this project and this report is to give an overview of the currently available literature on both the storage assignment problem and the order picking problem and their respective solution techniques. This is done in order to provide more in-depth analysis on the research topic before answering the final research question. In this report, the literature research is structured as follows. First, the research outline is given in Chapter 2. This mentions the research question and the structure of the full research plan. Afterwards, how the Picnic process of order picking works is explained in detail in Chapter 3. The existing literature on the online market of groceries is discussed in Chapter 4. Next, the most relevant exact solution techniques that can be used for the problem are laid out in Chapter 5. In Chapter 6 the literature on meta-heuristics used for similar problems is mentioned in detail.

2
Research Outline

This chapter gives the outline of the research on multiple storage locations in an e-grocery FC. Section 2.1 gives the full definition of the research problem. Afterwards, the objective of the research is given in Section 2.2. At the end, the main research question and the sub questions can be found in Section 2.3

2.1. Research Problem

The largest operation in FCs is the operation of order picking [5]. When order picking, order pickers walk from storage location to storage location, to retrieve all the number of items of products that make up multiple orders. The storage locations are combined in the picking circuit, which looks like your average grocery store: different aisles parallel to each other where shelves hold all the items. For Picnic, an online grocery retailer, the process of order picking and storage location assignment currently is a sequential process. First, items are allocated to storage locations and next the items are picked to fulfil the orders coming in. Each item only has one storage location in the picking circuit. Order pickers walk past every storage location to ensure that they are able to pick all the items needed. Storage locations of items are changing, due to seasonal trends (Easter eggs are only available from February to April for instance), demand differentiation and because of practical reasons. Practical reasons would be that the heavy items are placed first in the picking circuit, before lighter or more fragile items. This ensures that the heavy or robust items are first loaded into the client order and that they will not crush more fragile items, and it prevents the order picker from having to readjust the different items. Next to this fragility constraint, contaminating items such as cleaning products are often kept separate from fresh items.

The fact that all order pickers need to walk through the full pick circuit is, however, very inefficient. They are walking past every item, even though they only have to pick or collect a very small amount of items. If order pickers could skip part of the order picking circuit, their walking meters would be significantly reduced and subsequently the time spent per pick round would be reduced. This of course reduces the costs, if the same amount of orders can be picked in less amount of time. Yet, it also gives room for productivity increase. In the same amount of time, an order picker can then pick more orders than before. Skipping part of the circuit can only be done if items are strategically placed throughout the picking circuit and the routing is adjusted accordingly. This problem is called the *storage assignment problem (SAP)*. This research aims to solve the SAP, to be able to have more efficient routing in the FC. This is done by using multiple storage locations for the same item.

Using more than one storage location has been studied by Jiang et al. (2021) [9], where they make use of duplicate storage locations to minimise walking distance. However, in their studies, a storage location is considered to be a location for only one item, and it is only possible to have duplicate storage locations. Moreover, the routing that is considered in their paper is unconstrained, whilst there are often routing rules in FCs due to space restrictions as well as certain logic for the order pickers to follow. The research of this report is focused on storage locations that hold multiple items (of the same product), as well as having the possibility to use multiple storage locations for one item, instead of only duplication. This gives more freedom and possibilities, but also increases complexity. Yet, the routing is more constrained than in the previous research as well as the sequence of the picking of items due to the fragility and contamination as explained before. It is expected that optimising the SAP by using multiple storage locations and adjusting the routing of the order pickers accordingly, will significantly increase the efficiency of order picking activities, hence contributing to a much more efficient FC.

2.2. Research Objective

In the previous section the problem of the research was stated, and the explanation of why the problem currently exists. Having this in mind, the objective of the research is, as mentioned in Chapter 1, to improve storage location assignment by using multiple storage locations in an e-grocery fulfilment centre, such that the routing of order pickers can be done in the most efficient way in terms of time. To reach this objective, likely an exact model will be developed, which will as a benchmark for a meta-heuristic model to include more realistic scenarios. Assigning storage locations can be done in multiple ways, such as class-based storage, random storage or dedicated storage [7]. At Picnic previously class-based storage was used along with manual practical changes to the storage assignment. With around 7,000 different stock keeping units (SKUs) in the FCs, this will never yield an optimal solution. Hence, the goal is to develop a model that can provide an optimal solution. The model will be tested with historic data from Picnic on order picking activities and customer orders. With this data, a benchmark solution can be established, to later evaluate the model performance by comparison with the meta-heuristic model.

2.3. Research Question

The main research question has been defined as follows:

How can the time spent on order picking rounds be improved with regards to current operations by using multiple storage locations and optimal routing in an e-grocery fulfilment centre?

The usage of multiple storage locations to achieve more efficient routing has not been adapted in an e-grocery FC before, hence the research question is focused on this predicament. To answer the main research question, several sub-questions have been posed as well. The answers to these sub-questions will provide the answer to the main research question. The sub-questions can be found below.

1. **Which constraints need to be taken into account when assigning new storage locations and adjusting existing routing?**
   Within an e-grocery FC, especially within existing ones, there are certain constraints that need to be accounted for in order for the outcome of the model to be feasible and realistic. These constraints need to be determined in order to account for them. Constraints come into play when discussing items, as well as when discussing locations. Certain locations can only house pallet items for instance and certain items are considered fragile. Also, for the routing there are a lot of safety constraints and operational constraints which need to be determined.

2. **Which items need to be multiplied in which zones?**
   From an in-depth potential analysis it is expected that the items that need to be multiplied can be determined. It is not possible to duplicate or multiply all items, since there is a space constraint in terms of the building that houses the FC. Determining where these items need to be placed, before or after cut-off points, is the next step in reaching the desired efficiency. On which decision variables the selection is made needs to be determined and also correlation between items needs to be considered.

3. **What other parts of the fulfilment operations influence the efficiency and how can they be adapted accordingly?**
Other parts of fulfilment operations are for instance the batching of orders. Orders are batched together to also account for more efficient routing. The batching of orders can influence the order picking operation and this needs to be quantified. Once that is done, these operations can be altered to better fit the order picking operation and its turn also increase the overall efficiency. Also, replenishment of multiple storage location need to be considered, since an order picker will need to make more decisions during order picking, or they need to be decided for the order picker.

4. **How can congestion in the order picking process be avoided?**
Having multiple order pickers working together in the same pick circuit can cause congestion. Congestion is disastrous for efficiency and therefore must be avoided. Having a certain storage strategy and deciding when an order picker picks an item from either one or the other location could benefit the flow of order pickers and therefore must be investigated and analysed within the model.

2.4. Research Planning

In total, the project consists of four phases. These phases can be found below and are visualised in Figure 2.1. A more in-depth view of all different phases and exact time planning can be found in the Gantt-chart in Appendix A, Figure A.1.

1. **Phase 1**
Phase 1 consists of a literature study and creating a research plan. The outcome is a literature report as well as a report with a research plan. This phase is called the literature phase.
2. **Phase 2**
This phase consists of two parts, the model preparation and the model development, and is named the initial phase. This part focuses on sub-questions 1 and 2 and starts with a kick-off meeting with the thesis committee. The initial phase ends with a midterm meeting, halfway during the thesis. The performance thus far will be evaluated and feedback is given, to be taken care of in the subsequent phase.
3. **Phase 3**
Phase 3 focuses on sub-questions 3 and 4 and handles the model validation, results and conclusion. It is the final phase. This phase starts after the midterm meeting and ends with a green light meeting. This meeting is the final meeting before the graduation and gives room to present final results. Again, feedback is given to incorporate in the final version.
4. **Phase 4**
The last phase, phase 4 or graduation phase, handles the last feedback and improvements that needs to be made before the graduation day. There are no new developments of the model in this phase, only handling of previous matters.

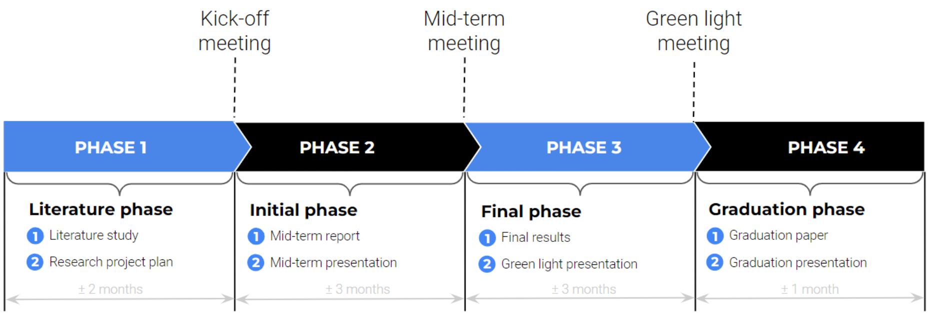


Figure 2.1: Thesis project planning.

3

The Picnic Supply Chain

This research is in collaboration with Picnic, a Dutch online grocery retailer with fierce competition, like mentioned in Chapter 1. Before the research discusses on different solution techniques, the Picnic supply chain will be explained, along with all the processes that take place in an FC of Picnic. First, the general and full overview Picnic supply chain will be given in Section 3.1. Afterwards, a deep dive into the technical systems and their components is given in Section 3.2. Lastly, in order to be able to measure performance, the key performance indicators (KPIs) of the Picnic supply chain are laid out and elaborated in Section 3.3.

3.1. Overview Supply Chain

Picnic has a relatively uncluttered supply chain and calls itself the modern milkman. The Picnic supply chain is unique in The Netherlands since it is specifically designed for e-groceries, unlike other supply chains for grocery stores which need to combine online and offline sales. Picnic does not have any physical stores and customers can only order groceries through an online application on their mobile devices. The supply chain consists of three parts:

1. Inbound process
2. Outbound process
3. Last-mile delivery

Both the inbound and outbound process take place at the FCs [10]. From the FC the groceries move to the hub via truck transport. The last mile delivery happens between the hubs and the customers and the transport of this phase is handled by electric Picnic vehicles (EPVs). An overview of the supply chain can be seen in Figure 3.1. Which activities belong to the inbound process and which to the outbound process is displayed in Figure 3.2. The inbound process starts with receiving: receiving items from the suppliers or distribution centres (DCs) at the docks. The suppliers are also pictured in Figure 3.1. Next, the put away process starts, which is the process of putting the received goods away at their storage locations or buffer locations (reserve locations). Replenishing consists of retrieving the items from their buffer location and storing them at their order pick location. From this location, the items are directly picked by order pickers, which is not the case for buffer locations. However, for Picnic it is often the case that put away and replenishment are the same process, because products do not always move

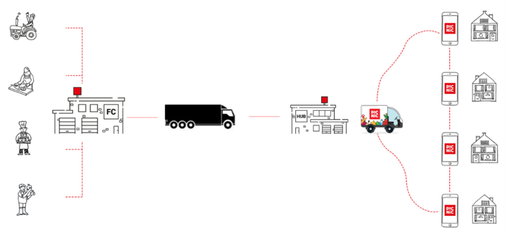


Figure 3.1: An overview of the Picnic supply chain.

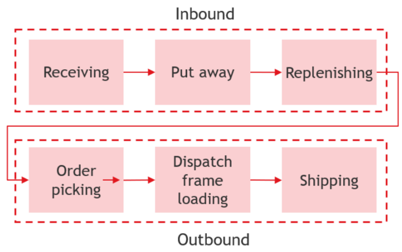
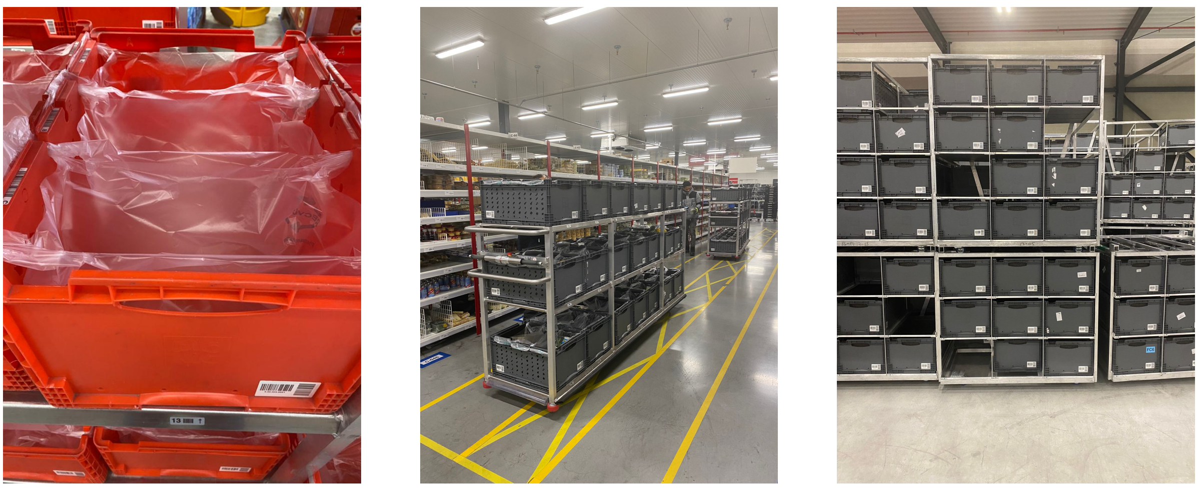


Figure 3.2: Inbound and outbound processes.

to a buffer location but directly to their order pick location. Afterwards, the outbound process begins. The main part of the outbound process is order picking, the collecting of items to fulfil customer orders. Once the orders are collected, they are loaded into dispatch frames which is called the dispatching frame loading. The dispatch frames are loaded into trucks, after which shipping to the hubs can take place.

During the processes within the FC there are several pieces of equipment which are used by order pickers. Some of these can be seen in Figure 3.3. These types of equipment are explained below:

- **Tote:** a crate filled with groceries, which is used for storing the ordered products after they are picked by the order picker. In one tote there are three different compartments, which all house one plastic bag in which the groceries are placed. Red totes are used for items in the ambient section (room-temperature items) and grey totes are used for items in the chilled section (fridge and freezer items).
- **Pick cart:** a motorised cart, which holds 21 totes. The order pickers steer these carts throughout the pick circuit.
- **Dispatch frame:** an aluminium frame that stores either 24 or 18 totes (depending on its height), which is used for the truck and EPV transport.
- **Roll containers (RCs):** high containers that are used to transport items in bulk from one place to another, such as from their buffer location to their order pick location.
- **Scanner:** an electric scanning device that tells the order picker which items to pick from which locations. The scanner is controlled by the Warehouse Management System (WMS).



(a) A tote with the three compartments **(b)** A motorised pick cart with 21 totes **(c)** Two different size dispatch frames

Figure 3.3: Equipment used by order pickers for the FC processes.

Next to the equipment used by order pickers, there is also equipment used to store all the groceries, to be seen in Figure 3.4. Within ambient and chilled there are three different kinds of storage equipment pieces:

- **Shelving units:** the most common storage equipment, consisting of shelves of different height and width, due to dividers that can be picked up and moved to a different place. Items are placed and stacked at the shelves and any boxes that group items together have already been removed.
- **Flow racks:** aluminium racks with wheels can house crates instead of single items. Especially suitable for fresh products such as fruits and vegetables, since they are often already delivered in these crates by the suppliers. Once an empty crate is removed at the front, the crate behind it automatically flows or glides to the front again.
- **Pallets:** a pallet consisting of a wooden or plastic base, designed to be easily transported with the help of (fork)lifts. They often house drinks or bulkier items and are stored without having the boxes or plastic around them removed.

(a) Shelving units **(b)** Flow racks **(c)** Pallets

Figure 3.4: Equipment used to store products within the FC.

The focus of this research is on the order picking process by adjusting storage locations. This adjustment has an effect on the receiving, put away and replenishment processes (inbound processes), however the changes needed on the inbound side are outside the scope of the research. Therefore, efficiency gains and results are also only considered for the outbound process of order picking. Whether the efficiency gain on the side of order picking outweighs the efficiency loss in the inbound processes (due to extra handling tasks), is interesting to look at from an operational and business point of view, but will not be discussed.

3.2. Technical Systems

Within Picnic there are several technical systems that have been developed in-house. Since Picnic likes to brand itself as a technical company, they store a lot of data and use algorithms to guide their processes. For the in- and outbound processes there are two very important systems: the Warehouse Management System (WMS) and the Master Planning Process (MPP). The MPP starts with planning after an order is placed, whilst the WMS starts being active last-minute before the order picking process. These systems and the algorithms that are used to build the systems need to be adjusted in order to account for having multiple storage locations and making the optimal use of them. Therefore, the systems will be explained, starting with the MPP in Section 3.2.1 and WMS in Section 3.2.2.

3.2.1. Master Planning Process

The MPP is made up of five sub-problems in total, which all have different objectives, such as minimising time or costs for instance. The processes are outlined below:

1. **Tote allocation service:** This service determines which items are placed into which totes. A tote has both a volume and a weight limit, which needs to be considered. The service is aimed at minimising the total number of totes and minimising the number of zones a tote travels through in the picking circuit.
2. **Dispatch frame allocation:** This algorithm ensures that a tote gets a place in a dispatch frame. The dispatch frames are filled with totes in the FC and two dispatch frames (ambient and chilled) fill up one EPV. This allocation needs to ensure that totes that are delivered by the same EPV end up in the same dispatch frame as well.

3. **Drop duration calculation:** The drop duration calculates how long a drop or a delivery will take. This takes into account the house type, for instance if the front door is at the ground floor or on the fifth floor, as well as the weight of the delivery and the number of totes.
4. **Trip planning:** The trip planning ensures that all deliveries are assigned to a certain trip with an EPV, while minimising the total number of EPVs and the time that is spent driving. It also assigns the available Runners (the employees who handle the deliveries) to the trips, taking availability break time and experience into account.
5. **Estimated time of arrival calculation:** The estimated time of arrival is calculated once all trips have been planned, taking the routing into consideration and the time slots that the customers have chosen to deliver their groceries.

For the research, problems two up to and including five are considered fixed and there are no changes needed. These problems are handled by the Distribution team of Picnic and do not influence the fulfilment processes. However, the tote allocation system is important when considering multiple storage locations, since this could give the problem more flexibility and hence yield higher results in terms of efficiency.

3.2.2. Warehouse Management System

The WMS, just like the MPP, consists of several sub-problems. The WMS handles all the processes that are taken care of in the FC, from small to big. Examples include the replenishing of items, receiving, creating buffers, keeping stock, guiding the order pickers, amongst others. Below, only the most important and biggest sub-problems are discussed.

1. **Slotting:** The slotting algorithm determines at which location an SKU is placed and how much space it gets. This is determined by certain picking categories and dividing those over the aisles of the pick circuit.
2. **Order picking:** The order picking service does not calculate anything itself, instead relays the information from previous tasks and algorithms to the order picker, such that the order pickers knows which items to grab and place into their respective totes.
3. **Tote to pick cart:** This algorithm decides which tote is placed on which pick cart. Having totes with similar items together on one pick cart reduces the amount of stops by the order pickers. Also, the number of zones the pick cart in its totality needs to traverse through is minimised.
4. **Tote position on pick cart:** This algorithm decides the divide between the 18 or 21 positions that are on a pick cart for all totes. The totes that have the most items are placed in the 'golden zone', such that the order pickers limit the number of times they need to bend down. Also, totes that share more similar items are placed close together.

For the research of the project at hand, sub-problems one to three are of importance. The slotting algorithm needs to be adjusted in order for it to be able to handle multiple storage locations for the same SKU. Order picking will be changed, since there are new routing possibilities of which the order pickers need to be informed about. Also, the tote to pick cart allocation will be altered, in order to have the totes with similar zones together on one pick cart.

3.3. Key Performance Indicators

For warehousing in general, as well as for Picnic, there is a big scope of KPIs. Within Picnic these KPIs are used to measure performance for each FC, hub or DC and to be able to compare performances. In order to have a scientifically significant result, it is important to establish which KPIs are of importance to the problem at hand and how they are calculated. Since this research is focused on optimising the performance of the order picking process, only the KPIs that influence this process are discussed. An overview of all the different KPIs can be found in Table 3.1. Below these KPIs are discussed and some further explanation about them is given. These KPIs are also used throughout literature, which is referenced accordingly. There are many more KPIs that are used to measure performance, yet the ones mentioned are the ones of importance to the research, since they all use a measure of time.

Table 3.1: Key performance indicators of significance to the order picking process.

KPI	Abbreviation	Unit
Seconds per item	SPI	Seconds
Units per hour	UPH	Units/hour
Picking productivity	PP	Order lines/hour

Seconds per item (SPI): This KPI gives the average handling time per item and is used to measure productivity. The time span starts when an item is scanned and ends when the next item is scanned.

Units per hour (UPH): UPH is also a productivity measure, and UPH can be expressed in SPI and vice versa. It gives the total number of units (items) that have been handled in one hour. The average UPH lies somewhere between 80 and 120 points for FCs that are at full capacity.

Picking productivity (PP): The picking productivity is measured in the number of order lines per hour. This differs from the UPH, since UPH takes single items into account, whilst one order line can contain several items. Hence, the PP tends to be lower than the UPH.

4 Literature on E-Groceries

E-groceries have been a fairly new topic in literature, since it has only existed for the last twenty to thirty years, along with the rise of the internet [11]. However, during the most recent years online grocery shopping has been flourishing, and the interest from the people has been increasing, with surveys showing more than half of the people participating are willing to order groceries online in the future [12]. With the increase in e-grocery shopping, the literature has also been growing, which will be discussed in this chapter. First, Section 4.1 mentions the definition of e-groceries and what falls under this umbrella term. Next, Section 4.2 describes the common problems that researchers encounter when investigating e-groceries and what they entail.

4.1. E-Groceries in General

The concept of e-groceries has been adapted in many countries worldwide, especially in North-America, Asia-Pacific and Europe [13]. There are many terms that refer to the same thing as e-groceries, such as e-retail, e-commerce grocery shopping, online retailing or online grocery shopping, amongst others. In this report, the term e-groceries is used throughout, and it is defined as ”the sale of grocery goods via the internet or other online channels, for personal or household use by consumers” [14]. Hence, this is a business-to-consumer (B2C) concept. Any groceries that are sold to businesses online (B2B), are not included in the definition of the term e-groceries. The ’other online channels’ that are mentioned in the definition include mobile applications, which is the use case for Picnic as well. Note that the term only includes the sale of grocery goods, meaning that whether the goods are delivered to the consumers’ house or are being picked-up (click-and-collect) does not matter. This definition is broader than the one set by Kolesar and Galbraith [15], where it is mentioned that e-groceries need to check three boxes: (i) having an online product search function, which entails detailed information on all the products; (ii) having an online purchase function; and (iii) the capability to deliver products to distribute the products to the consumers. Hence, the third condition is not necessary for the definition of e-groceries as laid out in this report.

Table 4.1: Advantages and disadvantages for e-grocery retailers compared to traditional grocery retailers.

Advantages	Disadvantages
Unlimited trading hours	Bigger workload
Faster transactions	More logistic processes
Shortened product cycles	Lower margins
Enhanced customer service	

There are several advantages and disadvantages to e-groceries compared to traditional or regular grocery shopping, both on the retailer and the consumer side. Some of these advantages and disadvantages hold for online retailing in general, whilst some are only relevant for the e-grocery market.

The most important advantages and disadvantages for the retailer can be found in Table 4.1, whereas the same from the consumer perspective are found in Table 4.2.

Whilst there is the possibility to process more sales for e-groceries due to unlimited trading hours [16], this brings a bigger workload for the retailer. Not only because of the bigger volume that is possible, also because one sale for a traditional retailer requires less work than one sale for an e-grocery retailer. This is because the products are packed by the company instead of the consumer and they often need to be delivered as well. These extra processes require more logistic operations in the supply chain and therefore can become more complicated quite quickly [17]. However, since there is much more tracking of the processes used to fulfil the customer order, the customer service is often found to be better [18].

Table 4.2: Advantages and disadvantages for e-grocery consumers compared to traditional grocery consumers.

Advantages	**Disadvantages**
Convenience	No immediate delivery
Time savings	Delivery costs
Home delivery	Lack of control for perishable goods
Economic value	Inadequate or no substitutions for out-of-stock items

The main advantage that consumers mention is the convenience that e-groceries bring [13]. A consumer does not have to travel to the store, spend time walking around and checking whether items are in stock or not. Everything is handled by the e-grocer. However, it is needed to plan in advance, since a lot of e-groceries have a minimum order amount such that you cannot order a single or a few items. Besides, it cannot be immediately delivered [19]. For Picnic, that is the reason why families are their target audience, since these usually plan ahead and have less time due to balancing work and family life [20]. Even so, the e-grocery market continues to grow, which indicates that for a lot of consumers the benefits outweigh the drawbacks [21].

There have been a number of literature studies on e-groceries and operational processes within these warehouses, such as Martín et al. (2019), Gong et al. (2011), Gu et al. (2010), De Koster et al. (2007), Cormier et al. (1992), amongst others [22] [23] [24] [5] [25]. Martín et al. describe the different research topics that there are in e-groceries and the current trends via their literature review. This review shows that the total number of papers or different documents on the topic have grown significantly since 1992, with the most popular topics being inventory management, last-mile delivery, return management and also the (beneficial) environmental effects of e-groceries. However, gaps still exist predominantly in the section of consumer behaviour and experimental design [22].

4.2. Common Problems in E-Groceries

There are quite some problems within the e-groceries research domain that are popular, or have been researched more often. However, the researchers have used a different approach to the same problem, or the general problem is the same whilst there are some differences in constraints. The most prominent problems that are found in the literature are mentioned below in random order and will be discussed in the subsections afterwards. This research focuses mainly on the storage assignment problem (SAP) and the order picker routing (OPR), however it cannot be completely separated from the other problems. It holds that researchers often look into a combination of the mentioned problems, in order to increase positive effects and tackle multiple things at once, and since the problems can be connected throughout the supply chain.

1. Storage Assignment Problem (SAP)
2. Last-Mile Delivery Problem (LMDP)
3. Order Picker Routing (OPR)
4. Order Batching Problem (OBP)

4.2.1. Storage Assignment Problem (SAP)

The SAP is a problem that occurs not only in the e-grocery market, but also in any other e-retailer market. However, the difference is that for e-groceries the number of storage locations is often bigger, items are much more volatile in terms of popularity and there is a high correlation between certain items. The definition of the SAP as set by Reyes et al. (2018) is: ”the allocation of products into a storage space and optimisation of the material handling costs or storage space utilisation” [7]. It is classified as an NP-hard problem, because of the large number of products and the different FC storage characteristics [26]. NP-hard stands for non-deterministic polynomial-time hard and is called as such if every problem in NP reduces to it [27]. It holds for these problems that there are no polynomial-time algorithms yet discovered that are capable of solving them, since the solution space is sized exponentially [28].

There are several approaches to the SAP, which can be a mathematical approach or not. The solution methods most often found in literature include exact methods, storage policies and rules, heuristics, meta-heuristics and simulation. For exact formulations, the most often used approach is through a MILP formulation, of which the model developed by Le-Duc et al. (2005) is an example [29]. Their goal is to minimise the travel distance of a pick round by solving the SAP, whilst the problem is constrained by five constraints in total. The first is about the length of the aisle that cannot be exceeded. The second is about the storage space for each class (group of items) that cannot be exceeded. The third set of constraints is about the relationship between the probability of an item being placed in a certain aisle and the length of the aisle. The fourth constraint is focused on the layout of the picking circuit, ensuring that it remains symmetrical since they are using a parallel aisle FC, that has a cross-aisle as well. Lastly, the fifth constraint is to ensure that the decision variables of the length of an aisle is non-negative. It can be seen that this is a MILP, since linear functions are used. Even so, the decision variables are not constrained to be integers. Solving such a model can be done through programming or commercial solvers, such as Gurobi or CPLEX [30] [31]. There is a certain linear objective to adhere to that either needs to maximised or minimised, however the constraints ensure that a solution is only possible if it is within the feasible space.

Another way to solve a problem that is formulated as a MILP is through heuristics or meta-heuristics. Heuristics and meta-heuristics might seem very similar, and they do share some similarities. Instead of searching for an exact solution, the solution that these methods offer is an approximation of the exact solution [32]. The best heuristics or meta-heuristics will give an answer to the problem that is as close to the exact solution as possible. However, there is one big difference between regular heuristics and meta-heuristics. Heuristics are problem specific, whilst the same meta-heuristic can be applied to multiple problems [33]. An example of a heuristic that was developed is in a paper by Battini et al. (2015), where they offer a decision-making process for both the manual picker-to-parts system and the storage assignment [34]. They have developed a heuristic titled the storage assignment and travel distance estimation (SA&TDE) joint method. This heuristic can be used at different ’levels’ in the FC, such as on area (multiple aisles) level or on aisle level. A flowchart of how this heuristic work is shown in Figure 4.1, along with the corresponding mathematical model.

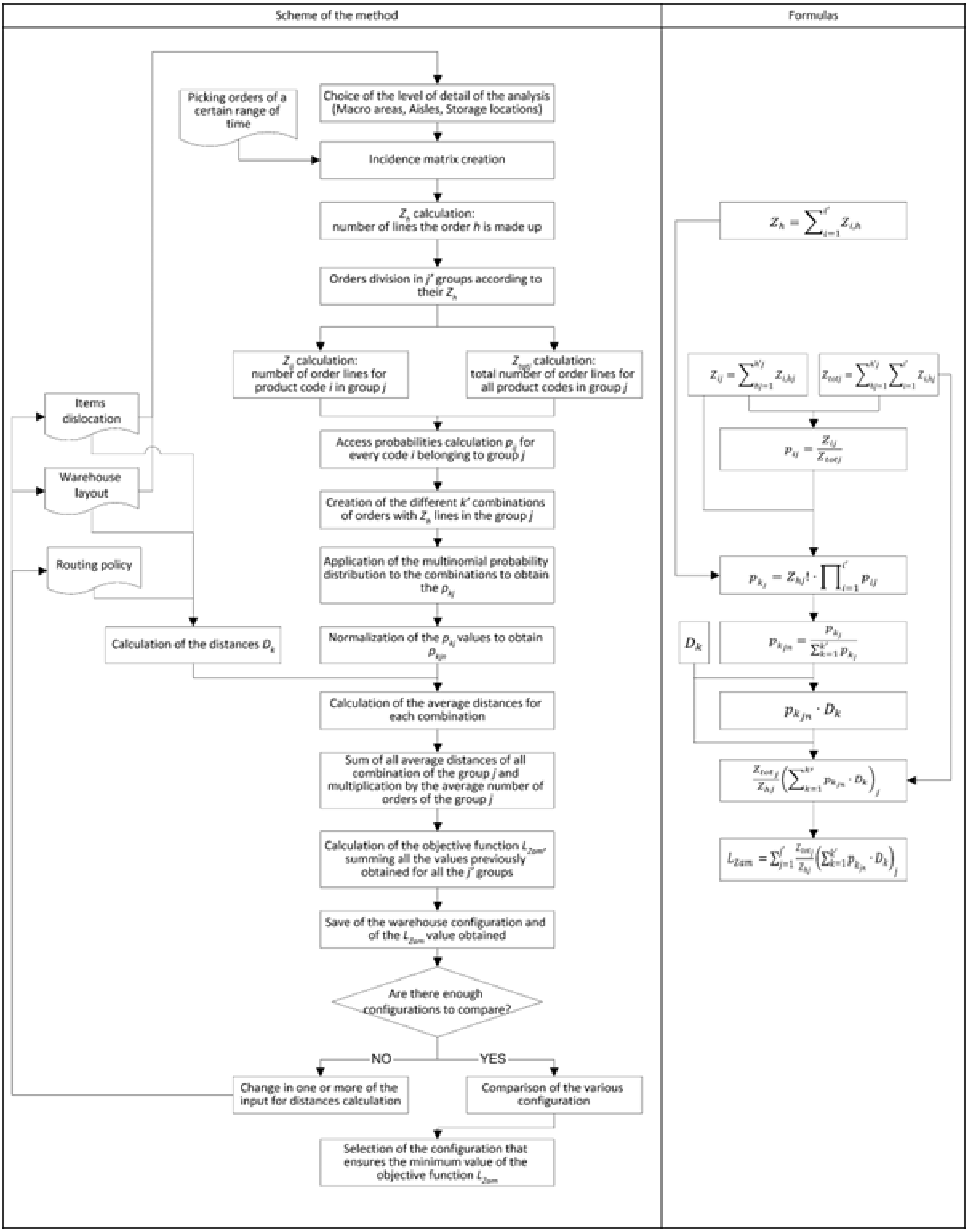


Figure 4.1: SA&TDE joint procedure with multinomial probability distribution flowchart [34].

Often used meta-heuristics include tabu search and genetic algorithms [7]. One example of a tabu search is one by Chen et al. (2010), in which they use this meta-heuristic to tackle the relocation of items, which means moving an item from one storage location to another storage location [35]. A tabu search begins with an initial solution and iterates from this solution to another until a certain termination criterion is met. It searches within the neighbourhood of the current solution and each iteration or operation is called a move. However, besides this, the algorithm also has a short- and long-term memory in which the tabus are stored, to avoid visiting the same solution twice. A tabu is a prohibited move [36]. This affects the computational time that is needed to solve the problem, since the same solution will not be visited over and over again. More information about tabu searches is given in Section 6.1. The tabu search algorithm employed by Chen et al. is summarised as follows:

Step 1: Initialise the problem through an initial solution and an empty tabu list.

Step 2: Apply a neighbourhood search strategy, determine the set of swap operations and select the best non-tabu move. With this a new neighbourhood is determined and a new tabu list, which serves as the starting solution for the next iteration.

Step 3: If the relocation time is smaller than the previous relocation time, update the solution, reset the iteration counter for improved solution and repeat from step 2.

Step 4: Terminate the algorithm if there is no improvement of the best solution after a set number of iterations, or if the total number of iteration reaches a maximum, predetermined value.

Approaches to the SAP that do not involve mathematics are storage policies and rules. An example of storage policies and rules is the class-based storage, as used by Fontana and Cavalcante (2014) [37]. Other policies and rules include random-based storage, dedicated storage or closest-open location, amongst others [7]. Fontana and Cavalcante use class-based storage to solve the SAP, meaning that all items that are housed in the FC are divided into classes and each class has a designated storage area. Which storage area a class is assigned to can be determined by some characteristics that the class might share. For example, the cube-per-order index (COI) can be defined per class of products, after which the designated space for each class is based on their COI. Belkaid et al. (2010) also use class-based storage, based on the total demand of an item, for which they have used an ABC-analysis [38]. Dependent on what the characteristics of the FC are and what is needed from an operational point of view, storage policies and rules can be a very efficient way to tackle the SAP.

Lastly, simulations can be used to tackle the SAP. Simulations can be present in many forms, but one that was used for the SAP is agent-based simulation, for example by Elbert and Müller (2017) [39]. Here, the agent visits every location from their order in the order that is determined by the routing policy, with the pick cart as means of transportation. The time of this pick round is subsequently calculated, and 1000 different tours were simulated using different routing and different storage. From these simulations, comparisons and conclusions based on the time as set by the simulation were drawn, showing that turn manoeuvres can heavily influence the operating strategy regarding routing, especially in smaller FCs. Hence, assigning storage as to minimise turns might have a beneficial effect. Simulation is often used to compare different types of policies or storage methods and draw conclusions based on the time spent on completing orders of customers.

4.2.2. Last-Mile Delivery Problem (LMDP)

Within e-groceries that offer home delivery services, the LMDP is a problem that has been studied very often. It is a classic case of a vehicle routing problem (VRP) and is considered to be very important to the e-grocery business, since it is the only direct interaction between the company and the consumer [40]. The LMDP handles the problem of given a depot, or hub, a set of vehicles, or EPVs, and their characteristics, and a set of customers, what is the most efficient routing to deliver all orders at the corresponding customers? This problem is related to the other operational problems, like the order batching problem discussed in Section 4.2.4, because it determines which orders need to be grouped together for delivery. Next to that, it determines at what time orders will be picked and hence which orders have higher priority in the fulfilment operations due to the delivery window the retailing company needs to adhere to.

According to a literature study done by Mangiaracina et al. (2019), there are three different perspectives to approach the LMDP found in literature: (1) environmental sustainability perspective; (2) effectiveness perspective on a customer service level; and (3) the efficiency perspective in terms of costs [41]. The focus in this report is on the efficiency perspective, since that is the perspective that approaches the problem as a VRP. The LMDP is also classified as an NP-hard problem and therefore has similar solution methods as the SAP, where the focus is on finding the optimal route to deliver all the orders to the customer, given constraints such as the range and capacity of the delivery vehicle [42]. A visualisation of the LMDP/VRP can be found in Figure 4.2, where the depot (d_1) can be considered to be a hub of an e-grocer and the trucks depicted as EPVs, in the case of Picnic.

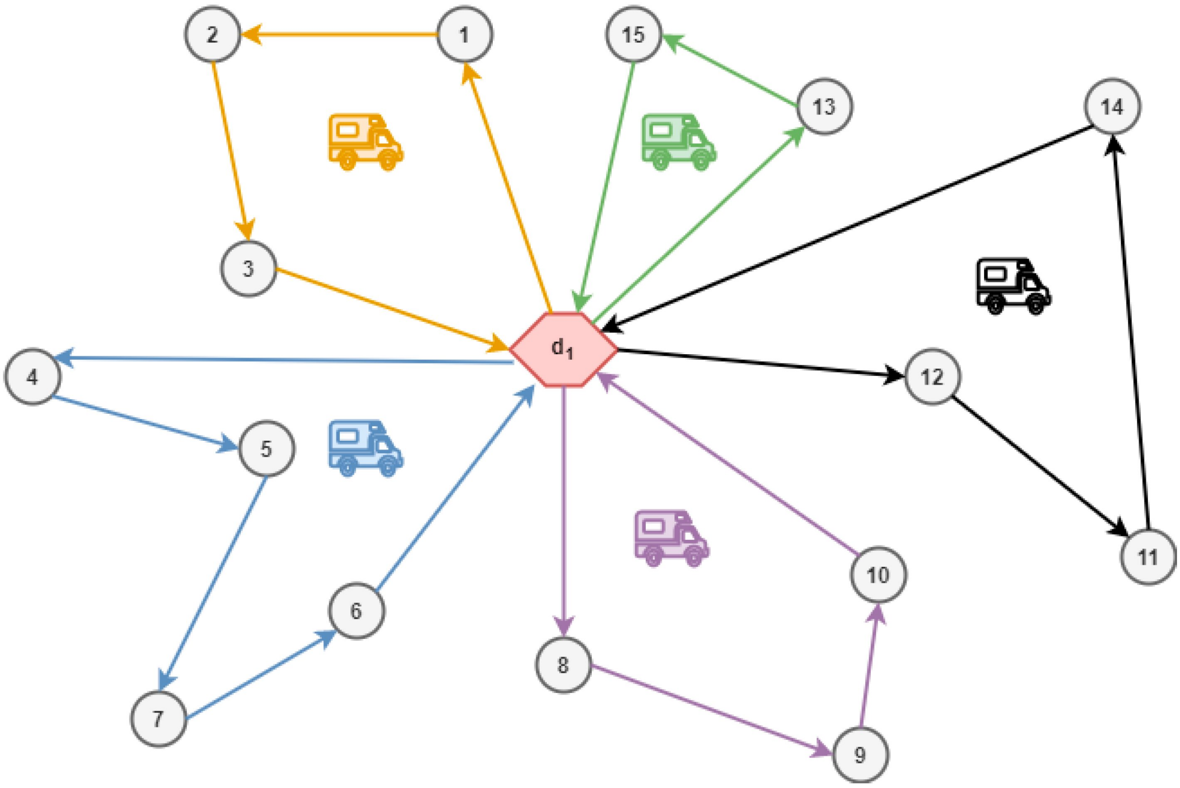


Figure 4.2: Visualisation of a vehicle routing problem [43].

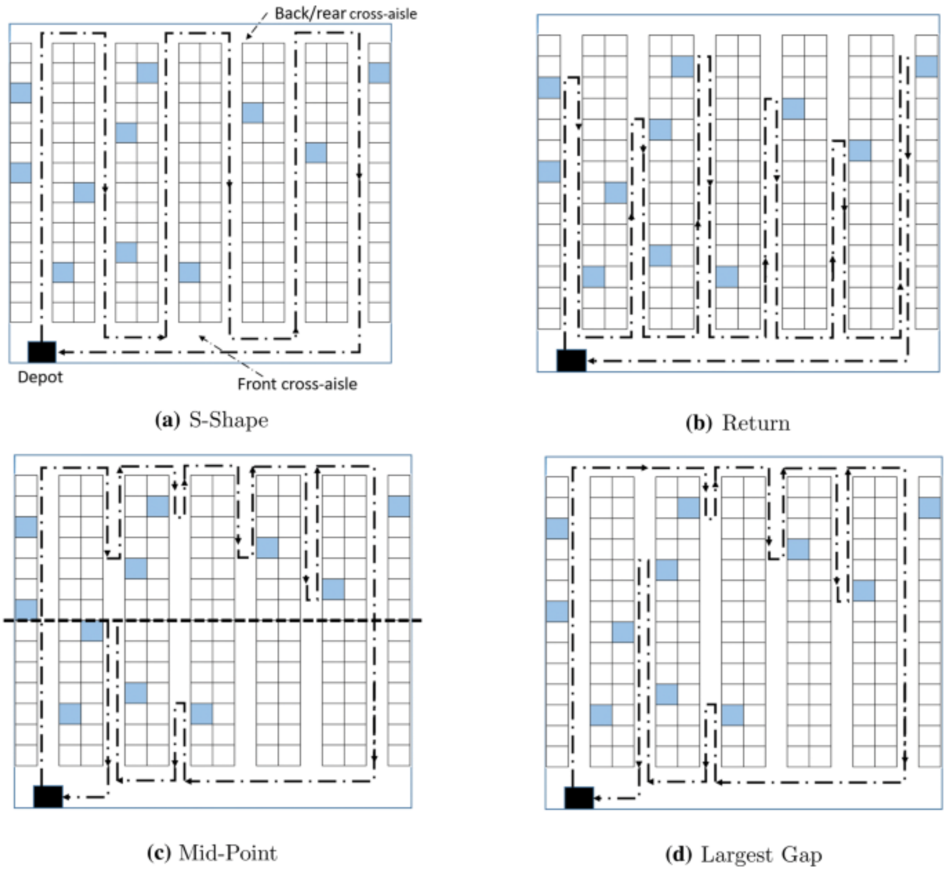
There are many examples in literature that solve the LMDP for e-groceries, such as Leyerer et al. (2020), who look at new ideas to minimise the costs, or Waitz et al. (2018), which also take the perishable products into consideration [44] [45]. Even though the LMDP is straightforward as to what kind of problem it is and how it can be solved (also through exact methods, heuristics, meta-heuristics or simulations), there are still innovative ways to tackle the problem and increase the efficiency. Some of these solution includes reception boxes installed at either public places or at houses, using drones or robots (self-driving vehicles) or making use of underground delivery [46] [47] [48] [49].

4.2.3. Order Picker Routing (OPR)

Since order picking is the most labour-intensive process in an FC, this has been a field of interest for a lot of researchers [50]. For this research, the focus is on the picker-to-parts system, where the order picker walks along the aisles to pick the items of orders. This is also the most common used system, since it requires less investment costs than an automated system for instance [51]. The focus is also only on low-level picking, meaning that the items are stored at a low level and the order pickers are able to grab the items whilst standing. Contrary to low-level picking, high-level picking uses high storage racks, for which a crane or a lift is needed to reach all the items [5]. The OPR problem handles the problem of given a set of orders to retrieve and their location, what is the most efficient routing to retrieve all orders from the corresponding storage locations.

This is another version of the travelling salesman problem (TSP), which is formulated in the same way: ”given a set of cities and the distance of travel between each possible pair, the TSP is to find the best possible way of visiting all the cities and returning to the starting point that minimise the travel distance (or cost)” [52]. However, there still remain a couple of differences between OPR and the TSP, since for OPR there are a number of nodes that do not have to be visited, but still can be visited. Subsequently the nodes that do need to be visited, such as the pick locations, are allowed to be visited more than once. Therefore, the OPR can be classified as a Steiner TSP, which is generally not solvable in polynomial time [53]. However, there have since been multiple algorithms that are able to solve the problem, such as De Koster et al. (1998) [54].

Most often OPR is solved by using heuristics, since there are operational difficulties when using optimal routing. Optimal routing may not always be the most logic route for the order pickers, which can make them deviate from the route [55]. Also, congestion is not taken into account by using optimal routing even though it can significantly impact the efficiency of the routing [56]. An example of a heuristic used to tackle OPR is one developed by Chen et al. (2019), in which they made a routing model to minimise the travel distance of picking an order and subsequently minimise the time spent on order picking [57]. They develop multiple heuristics based on existing heuristics and compare these, to determine which would work best in a narrow-aisle FC that also has access restrictions regarding the picking carts. Their paper illustrates that there are multiple solutions to the problem, and it is dependent on the objective to rank the solutions. Widely-spread heuristic methods for OPR are the S-shape method, return method, the mid-point method and the largest gap method. The first two methods are both studied by De Koster et al. (2007). The S-shape method is widely used and is one of the simplest heuristics, which sees the order pickers walking one-way and for any aisle that contains a pick location the complete aisle is traversed. The return method entails that the order picker both enters and leaves each aisle from the same end [5]. The mid-point method uses the mid-point of the order picking circuit and divides the area of the circuit in two halves. Picks are accessed either via the back or front cross aisle, depending on whether the pick is located in the half of the back cross aisle or in the half of the front cross aisle [56]. The largest gap method is a strategy that sees the order picker walking from both sides of the aisles, where the largest gap is the portion of the aisle that the order picker does not cross through, whilst still entering and exiting an aisle from the same side [58]. Visualisations of these four heuristics can be seen in Figure 4.3.



(a) S-Shape (b) Return (c) Mid-Point (d) Largest Gap

Figure 4.3: Four different heuristics used for order picker routing [59].

The OPR is also something that can easily be simulated, as has been done often such as by Shetty et al. (2020). They show that significant improvements can be made from the traditional heuristic approaches by implementing the best practices from simulation [59]. Another application for simulation with OPR is for researching congestion, which is an important factor in the OPR and its success [60]. One of the remaining important difficulties with OPR is that there still is a human factor that needs to be taken into consideration. Humans can make the decision to deviate from the route that is laid out for them and hence follow a suboptimal route. This has been studied by Elbert et al. (2017) also through simulation, from which it is concluded that route deviations often occur in practice, with the main reason being that the optimal route seems illogical to the order pickers. This significantly affects the performance of the routing policies and its efficiency [61].

4.2.4. Order Batching Problem (OBP)

The OBP is an important problem in terms of impact in the FCs. The solution to the OBP determines which orders are batched together, in order to increase efficiency. In the case of Picnic for instance 21 totes are placed together at one pick cart, which can be considered to be one batch. The OBP needs to decide which totes belong to one pick cart, such that the productivity of the order pickers is maximised. It needs to be the case that the tour that is needed to retrieve all items for the batch of orders is shorter than the tours that are needed to retrieve all orders separately [62]. This entails that again the objective is to minimise total distance and subsequently time. The OBP is classified as a clustered TSP or a capacitated VRP, since the order batches can be considered clusters and the pick locations as cities, which need to be visited contiguously [63].

Again, this problem can be solved through exact methods, as has been done by Bozer and Kile (2008), who formulated the problem as a MILP and found optimal solutions for smaller instances [64]. However, they also took a look at heuristic approaches, such as the first fit-envelope based batching, developed by Ruben and Jacobs (1999) [65]. In this heuristic each order gets an index based on the minimum and maximum aisle number visited. The orders are sorted by this index and batched together in the same manner, until the capacity of the batch (or pick cart) is reached. Other heuristics also exist, some already developed last millennium, which have a decent performance. Another exact approach is done by Gademann and van de Velde (2005). They present a branch & price algorithm for the OBP which they solve through column generation [55]. However, this model can also only provide optimal solutions for very small instances, which are not close to real instances. This tends to be the case for exact solutions to the OBP [66]. Meta-heuristics that can be used to tackle the OBP are ant colony optimisation, tabu search and genetic algorithms amongst others [67] [68] [69]. Simulation methods seem to be less used for the OBP than for the other problems described.

5

Exact Solution Methods

Like mentioned, there are multiple solutions to the SAP and OPR. These solutions can be either exact or (meta-)heuristics are used, to improve computational time. This chapter focuses on the different type of exact solution methods there are to a MILP, since both the problems can be formulated to a MILP and this is the most common formulation in existing literature [7]. The objective functions and constraints are linear, whilst some variables are constrained to be integers and others can also be non-integers. There are many existing solution methods, yet the focus in this chapter is on the methods that are used most frequently. These are branch & bound, branch & price, branch & price & cut and column generation [70]. They will be discussed in Section 5.1, Section 5.2, Section 5.3 and Section 5.4 respectively. Afterwards, Section 5.5 mentions the different solvers that are available to solve the problem and their advantages and disadvantages.

5.1. Branch & Bound

Branch & bound is one of the most general approaches to find the solution of an optimisation problem, especially when it is constrained. Branch & bound is a way to structure the search of the feasible solution space [71]. A branch & bound method is often used for the TSP, hence it can be used for the OPR and OBP for instance [72]. However, this method is not limited to MILPs, it can be applied to a number of different integer programming problems. The branch & bound approach is based on the fact that the total set of feasible solutions can be divided into smaller sets of the solution, in subsets. Then, the smaller subsets are evaluated systematically, until the best solution is found [73]. This entails that the problem is structured in a clever manner, such that only part of the feasible solutions need to be examined instead of all feasible solutions. It consists of three processes, branching, bounding and fathoming. Branching starts with the relaxed linear programming solution (the integer restrictions are relaxed) as a node, from which further branches are 'built'.

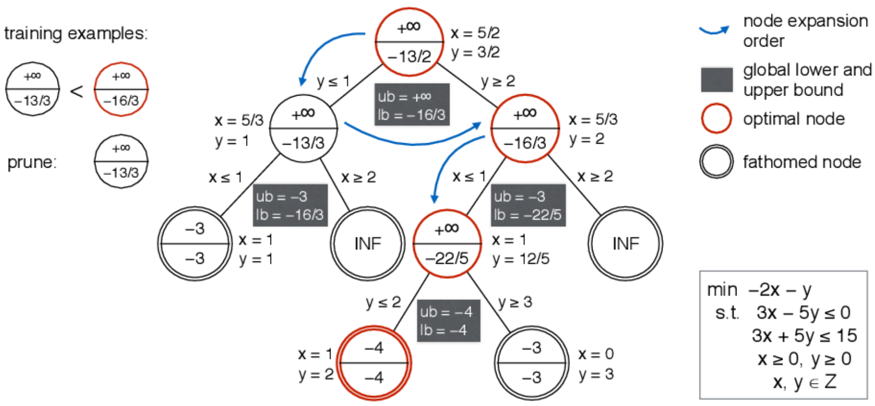


Figure 5.1: A visualisation of branch & bound method, including the nodes, branches and solutions [74].

Bounding is the process of limiting the solution space to an upper bound and a lower bound. For maximisation models, the upper bound is the value for the relaxed solution and the lower bound is the value for the rounded-down solution from the relaxed solution. The optimal solution needs to be in between these two bounds. From this, new branches are created within the two bounds. Branching keeps happening from the node with the maximum upper bound. This continues until there are no feasible solutions found, and one integer solution is the optimal one, being called the ending node. The same goes for a minimisation model, yet the relaxed solutions and upper- and lower bounds are reversed. Fathoming is the process of cutting of a branch and dismissing it from further consideration, if one of the fathoming tests is passed. A visualisation of how the branch & bound process works can be seen in Figure 5.1.

The branch & bound method is often used to solve the warehouse location problem in e-commerce, which handles the question about at which location to build a warehouse in order to satisfy customer demand in the most optimal manner [75]. However, there are also examples of using branch & bound algorithms for the OPR. The paper by Hahn and Scholz (2017) is one of these, where they use a truncated branch & bound algorithm, in order to minimise the time spent waiting for order pickers [76]. A truncated branch & bound algorithm still uses the branching approach, however a heuristic method is used to cut off certain branches to reduce the computational time needed to solve the problem [77]. Because of this, an optimal solution cannot be guaranteed to be the outcome. The model was tested and validated for 200 customer orders and with an S-shape strategy, after which the reduction of total waiting time could be determined. The average reduction of waiting time was 72.6%, which is a large improvement.

Another example of a branch & bound approach to e-grocery related topics is the research conducted by Muppani and Adil (2008), where they use the algorithm to determine the class based storage location assignment, a version of the SAP [78]. They split up the algorithm in three main steps, by first forming the search tree, subsequently computing the bounds and then tackling the process of fathoming. To test the algorithm, a distribution warehouse that houses 45 items is considered. By using the new algorithm and adjusting storage accordingly, costs are reduced by 10% whilst computational time is less than eight seconds. Therefore, the authors argue that their algorithm can handle large real life problems efficiently as well. This was not tested, yet the problem grows exponentially when adding more items. Selling only 45 items is not realistic when comparing it to current e-groceries and their assortment sizes.

The approach of a branch & bound algorithm is found more often in past literature than in recent literature, probably due to the rise of meta-heuristics and its performance. Singh and van Oudheusden (1997) used a branch & bound algorithm for the travelling purchaser problem (TPP) [79]. A general approach was used, where problem size, computational time and costs varied. Afterwards, the applications to real life situations are discussed, where they mention that their approach can also be used for the OPR problem as a special case of the TPP.

5.2. Branch & Price

Branch & price is used to solve integer linear programming models that have many variables. It is a method that combines the branch & bound technique mentioned in Section 5.1 and the column generation method, which will be discussed in Section 5.4. Barnhart et al. (1996) present a general methodology for the branch & price algorithm, by using the generalised assignment problem (GAP) and crew scheduling problem as examples [80]. For this solution technique, columns may be added to the relaxed linear programming model at each node of the branch & bound method. First, the problem is reformulated by using the Dantzig-Wolfe decomposition for example, to give the reduced master problem (RMP) and from this problem branches are created. More information about how the Dantzig-Wolfe decomposition works can be found in the work of Vanderbeck and Savelsbergh, amongst others [81]. Afterwards, columns are added to improve the previously found optimal solution, yet not all columns are needed since many decision variables will have a trivial solution. Which columns need to be added is determined by solving the pricing problem. The pricing problem identifies a column that has the maximum reduced cost. If this number is greater than zero, the column needs to enter. Otherwise, the solution at hand is the optimal solution.

The branch & price solution technique is often used when the relaxed model is very weak and the model cannot be solved using the branch & bound approach except for very small instances [82]. Hence, the Dantzig-Wolfe decomposition is used, and column generation is applied since the number of variables is so large. This entails that the branch & bound problem has been transferred into a branch & price problem. A flowchart of the technique to solve the location-routing problem with time windows, which is a version of the LMDP, can be found in Figure 5.2.

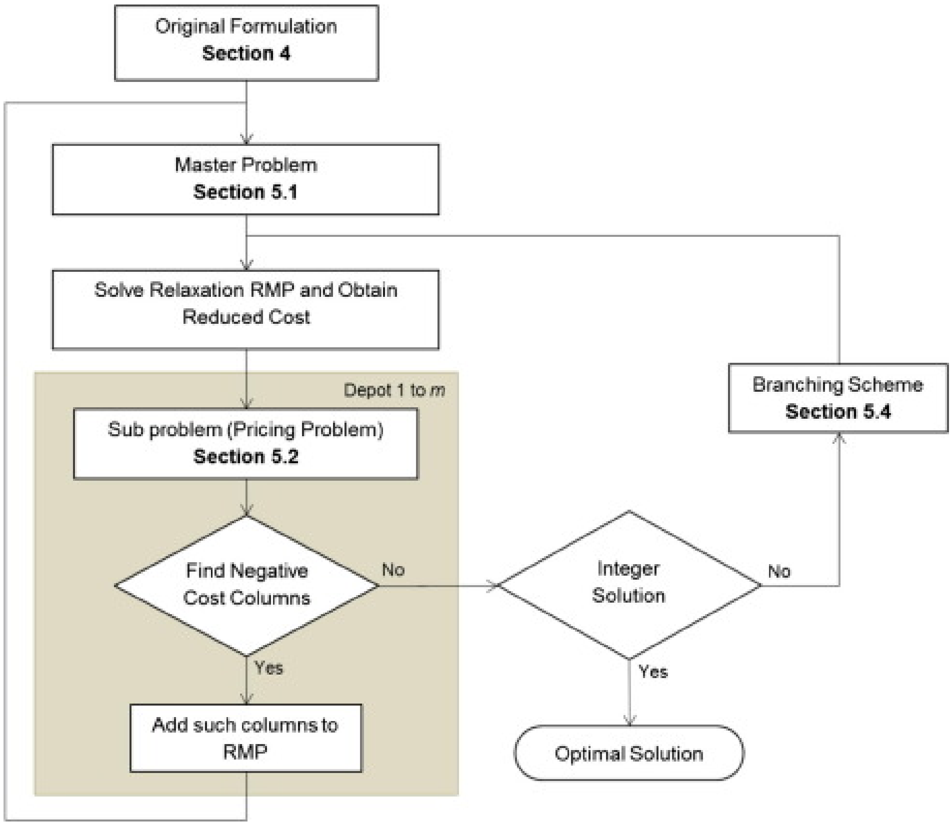


Figure 5.2: A flowchart of the branch & price method to solve the location-routing problem with time windows [83].

The branch & price solution technique is frequently applied to the GAP, like mentioned. The GAP is the problem of finding the minimum cost assignment of n jobs to m agents, such that each job is executed by exactly one agent. The SAP can be considered to be a version of the GAP, where a job is an item and an agent is a location. Examples of research on the GAP through branch & price include work from Ceselli et al. (2006), Firat et al. (2016) and Wang et al. (2021), amongst others [84] [85] [86]. However, when looking at the branch & price technique in e-groceries the most often use for it is to solve the LMDP and routing strategies in general. An example of this is the work by Azi et al. (2010), where they use the algorithm to solve a VRP with time windows and multiple use of vehicles [87]. They develop a search- and a branching strategy, and test it on different instances. This has different effects on costs, the number of iterations needed, driven distance and the computational time. However, the model is limited by the fact that the algorithm can only solve instances of 25 customers, and a few instances of 50 customers. Therefore, they conclude that using (meta-)heuristics are still the best way to solve these problems and determine strategies, since instances can be solved that are closer to real life.

A different problem was tackled by Gademann and van de Velde (2005), where they assess the OBP through a branch & price approach [55]. The objective is to minimise the order picking time. They develop all the five different components of a branch % price technique, the integer linear programming formulation, the approximation algorithm to form the initial solution, the pricing algorithm, the column generation algorithm and a strategy for the branching. These are discussed, and the results show for different instances that the algorithm gives a good performance both for the order picking time and for computational time needed. The authors strongly believe that their algorithm could also work on larger instances and yield similar performances.

5.3. Branch & Price & Cut

The branch & price & cut solution technique is again a mix of methods, this time between the branch & price method and the branch & cut method. Again, this technique can be used to solve linear programming models with a very high number of variables. However, it is noted that the problems are usually only solvable if the percentage of coefficients in the functional constraints that are zeros needs to be very big and therefore do not all have to be considered [73]. This technique relies on the method of using a cutting plane, or a cut. A cut is a new constraint that further reduces the feasible region for the relaxed model, without eliminating feasible solutions for the integer programming model. There are multiple techniques to come to relevant cutting planes. Once more, it holds that this technique is used to limit the search space and hence find the optimal solution quicker, as well as to obtain stronger relaxations [88]. A visualisation of how cutting planes work is found in Figure 5.3.

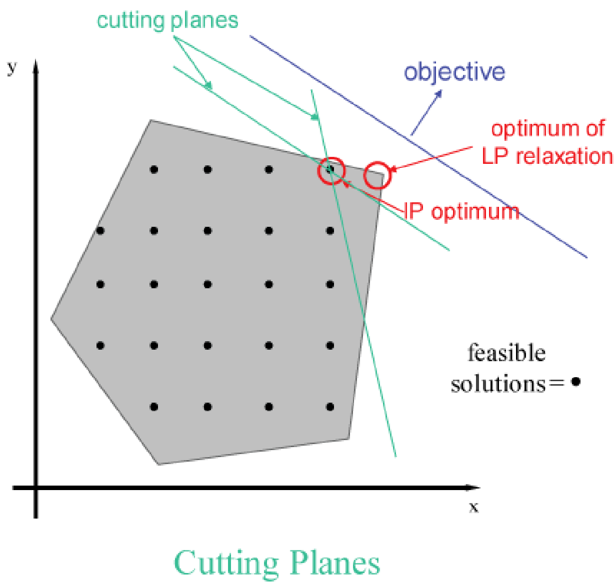


Figure 5.3: A visualisation of cutting planes [31].

Examples of this technique in the e-grocery business can be once again found most often as a solution to the VRP. Archetti et al. (2015) use the algorithm to solve the VRP where customers request multiple commodities [89]. Again, multiple instances were tested, and computational results were obtained for small and mid-size instances, up to 40 customers. In 50% of the instances they succeeded to improve or compute the best known upper bound values. However, like many, they state that for real life instances a heuristic should be applied. Another example of this is the paper by He et al. (2019), where they also use a branch & cut & price technique to solve the VRP with time windows, which is comparable with the LMDP [90].

For automated warehouses in e-groceries, the branch & cut & price method is often used for the multiagent pathfinding problem. This is the problem of defining the path for (automated) agents, such that they all reach their goal and avoid collisions with one another, whilst cost is minimised [91]. A good example of this is the algorithm from Lam et al. (2022), where they show that their approach reaches more solutions than state-of-the-art existing solvers used in the field of artificial intelligence [92]. They approach the problem through four steps: (i) creating a master problem, (ii) creating a pricing problem, (iii) adding separation problems as constraints and (iv) applying branching rules for their search tree. A lot of different conflicts are defined and worked out and implemented into the problem as constraints. Since the algorithm is of such high performance, the authors note that it would be interesting to see what the combination of communities from mathematical optimisation and artificial intelligence could yield in terms of scientific advances.

5.4. Column Generation

Column generation was introduced in 1958, by Ford and Fulkerson [93]. Same as for the other techniques mentioned, the key to this method is that not the entire search space has to be considered, only a part of it and is therefore especially suitable for large linear programming problems [94]. There are different methods within the column generation technique, yet all begin with defining an RMP. Which columns are selected to take part in the algorithm is what defines the differences between the methods. Each column is related to a decision variable and is added to the problem in case it yields higher results. Since a lot of the decision variables will be zero, not all decision variables need to be considered and hence the search space can be reduced. In this way, only decision variables are added that have a positive effect on the optimal solution. If there are no decision variables left that improve the current solution, then the optimal solution is reached [95]. A flowchart of a typical column generation technique is seen in Figure 5.4. Airline crew scheduling is an often used example for the column generation method, however job scheduling, vehicle routing and bin packing are also amongst the popular problems that can be solved with column generation [94].

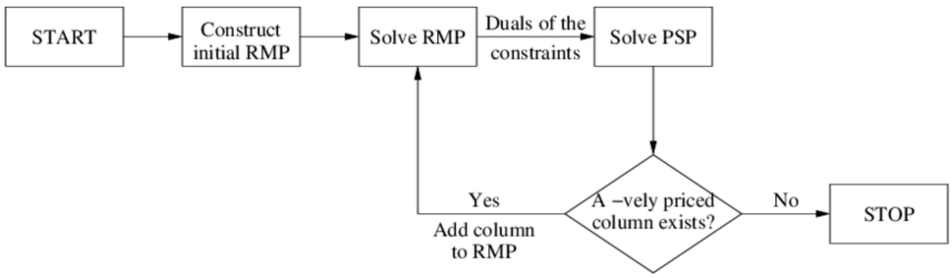


Figure 5.4: A flowchart of the column generation technique [96].

Column generation in e-groceries is often used to solve the VRP, but also as a base for certain heuristic. Examples of the latter include the work of Ardjmand et al. (2020), in which they develop a novel hybrid column generation heuristic, to minimise the makespan in manual order picking [97]. The heuristic is applied at the pricing step of column generation. Instead of solving the routing problem for order picking multiple times during each simulation, they use an estimation of the length of the route and solving the exact length at a later stage. By doing so, the column generation heuristic outperforms other tested against heuristics. However, since a heuristic is used it cannot be guaranteed that the optimal solution is found.

When looking solely at exact use of the column generation technique, the work of Moccia et al. (2008) studies the dynamic GAP through column generation [98]. It is mentioned that this problem had application in the management of warehouses when assigning storage to a certain location. A pricing routine needs to be determined to find the path with negative reduced costs. This is tested against certain heuristics that use the column generation as a base as well, which show that the heuristics outperform the exact solution technique.

Another application of the column generation technique is for the OBP and the scheduling of order pickers, which has been researched by Muter and Öncan (2021) [99]. They use the column generation technique to create order picker schedules, which correspond to a set of order batching, which both satisfy the operational time limits of the order pickers. This problem increases exponentially when the number of orders is increased. The objective is to decrease the amount of time it takes to collect all orders and to decrease the makespan of the orders. By comparing two column generation algorithms which prioritised one or the other of the two objectives. After computational experiments for all different routing policies considered, it was concluded that the order batching part of the problem is the non-trivial part, and they were able to find exact solutions for instances with up to 100 orders.

5.5. Commercial Solver Software

Solving MILPs by exact solution techniques is also possible through solvers. Solvers can be both commercial and non-commercial and they apply one of the algorithms mentioned in the previous sections, instead of a user having to create the entire model themselves. A comparison will be made between five solvers, CPLEX, Gurobi, LINDO, LP_SOLVE and SoPlex. An overview of these solvers and their characteristics can be found in Table 5.1.

Table 5.1: Solvers used for MILPs and their characteristics [70] [100].

Solver	Algorithms used	Features	Specifications	Modelling languages
CPLEX	Branch & cut algorithms and dynamic search algorithms	Capable of calculating multiple optimal solutions and have the solutions stored in a solution pool.	Version: 12.8.0 Website: http:// www.ibm.com /analytics/cplex-optimizer License: proprietary	C, C++, Java, .NET, MATLAB, Python, Microsoft Excel
GUROBI	Cutting planes algorithms, heuristics and search-techniques	MILP solver that is designed with modern multicore processing technology to obtain an optimal solution.	Version: 3.0 Website: www.gurobi.com License: proprietary	C++, Java, .NET, Python
LINDO	Different forms of cutting planes algorithms and different node selection rules	Significantly faster on large quadratic models. Improved handling of models with discontinuous functions.	Version: 10.0 Website: www.lindo.com License: proprietary	C, Visual Basic, MATLAB, Microsoft Excel
LP_SOLVE	Branch & bound algorithm	Advanced pricing using Devex and Steepest Edge for both primal and dual simplexes. Provides different scaling methods to make the model more numerical stable.	Version: 5.5.2.11 Website: http://lpsolve. sourceforge.net/5.5/ License: open source for academic use	Java, R, Python, MATLAB, Scilab, amongst others
SoPlex	Primal and dual revised simplex algorithm	Presolving, scaling, exploitation of sparsity, hot-starting from any regular basis.	Version: 6.0.0 Website: soplex.zib.de License: open source for academic use	C++

There has been multiple research done on the advantages and disadvantages of certain solvers and for which kind of problems one should use what solver. One example of this is the study done by Meindl and Templ (2012), where both the open-source and commercial solvers mentioned in Table 5.1 are assessed [100]. Usually, open-source solvers have their code available, which makes it easy to implement in other modelling languages or on different platforms. Even though every problem that is researched in the scientific world is different and has specific requirements where open-source solvers can be very useful, there has been far more research done with commercial solvers and in a lot of instances they outperform the open-source solvers.

For the problems at hand, the SAP and the OPR, there has been a lot of research done with commercial solvers. For both there are instances where CPLEX was used and instances where GUROBI was used. Weidinger and Boysen often use GUROBI in their research to find optimal solutions, for both exact techniques as well as for meta-heuristics [101]. They prove that for large instances GUROBI is still able to have an acceptable computational time. However, the same can be said for CPLEX, which can also be used to solve for optimal solutions. This is also often done to find intermediate solutions for larger problem sizes, since these solvers do not have enough memory to give the exact solution for very big instances [102]. Both these solvers can be used in Python which is the preferred modelling language for this research. This cannot be said for LINDO.

6
Meta-heuristic Methods

Next to the exact methods to MILP formulations, there are also a plethora of meta-heuristic solution techniques to solve the SAP and OPR. Dependent on their quality, they come very close to the optimal solution. Since they use their memory or different techniques, they do not have to evaluate the full solution space, they cut some corners and hence are much faster computationally than exact methods. As mentioned before, meta-heuristics can be applied to multiple problems, instead of heuristics that are problem-specific. Even though there are many meta-heuristics, this chapter focuses only on the most frequently used ones. These are tabu search, discussed in Section 6.1, genetic algorithms, discussed in Section 6.2, ant colony optimisation, discussed in Section 6.3, and large neighbourhood search, discussed in Section 6.4.

6.1. Tabu Search

The meta-heuristic technique of tabu searching was developed by Fred Glover in 1986 as a way of solving integer programming problems [103]. Tabu is another word for prohibited, and a tabu search thus prohibits certain moves from solution to solution being made [104]. It is mentioned that one of the pitfalls of heuristics can be to get stuck at local optima [105]. This, of course, needs to be avoided and hence, Glover tried to find a method that escapes the local optima. Tabu search is an extension of another meta-heuristic of local neighbourhood searches. Tabu search improves the performance of local neighbourhood searches by being able to accept moves that do not bring an improvement if there is no improving move as an option. It has three different types of memory to search through and base decisions on, (i) short term memory: the list of solutions that have been considered recently which include certain tabu moves and is used for exploration, (ii) intermediate term memory: includes intensification rules, and lastly (iii) long term memory: includes diversification rules [106]. Keeping this in mind, the general steps of a tabu search algorithm are as defined below.

Step 1: Determine a starting solution.
Step 2: Create a candidate list of moves from the starting solution to neighbouring options.
Step 3: Choose the best candidate from the list and make it the new solution. The best candidate is the most improving candidate, or if there are no improving candidates the least worsening one.
Step 4: Evaluate the stopping criterion and stop if a certain number of iterations has been met, if continuing update tabu list and criteria and repeat from step 2.
Step 5: Stop algorithm or transfer through the intermediate or long term memory, to initiate intensification or diversification.

This process can also be seen in the flowchart of Figure 6.1. As mentioned, mainly the diversification phase is what makes a tabu search an often used option and is one of the big advantages. A variable can be set to determine the amount of time spent diversifying. However, one of the main disadvantages of tabu searches is that it cannot guarantee convergence, and it is very dependent on the chosen initial solution [107]. Therefore, researchers continue to work on the tabu search algorithm and try to improve it.

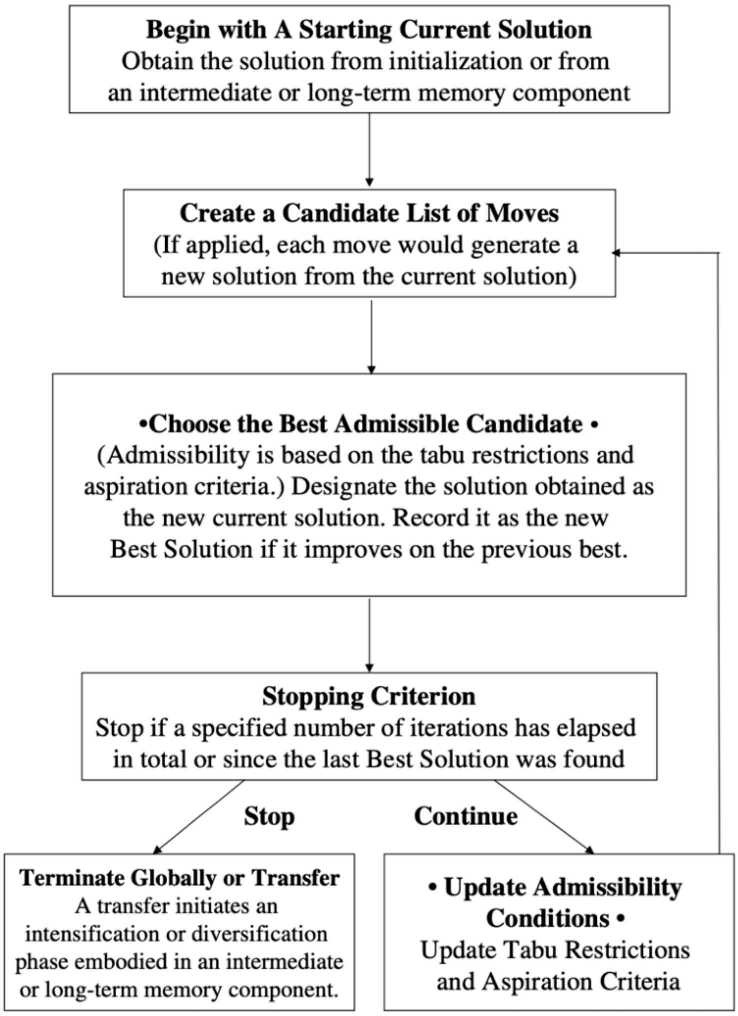


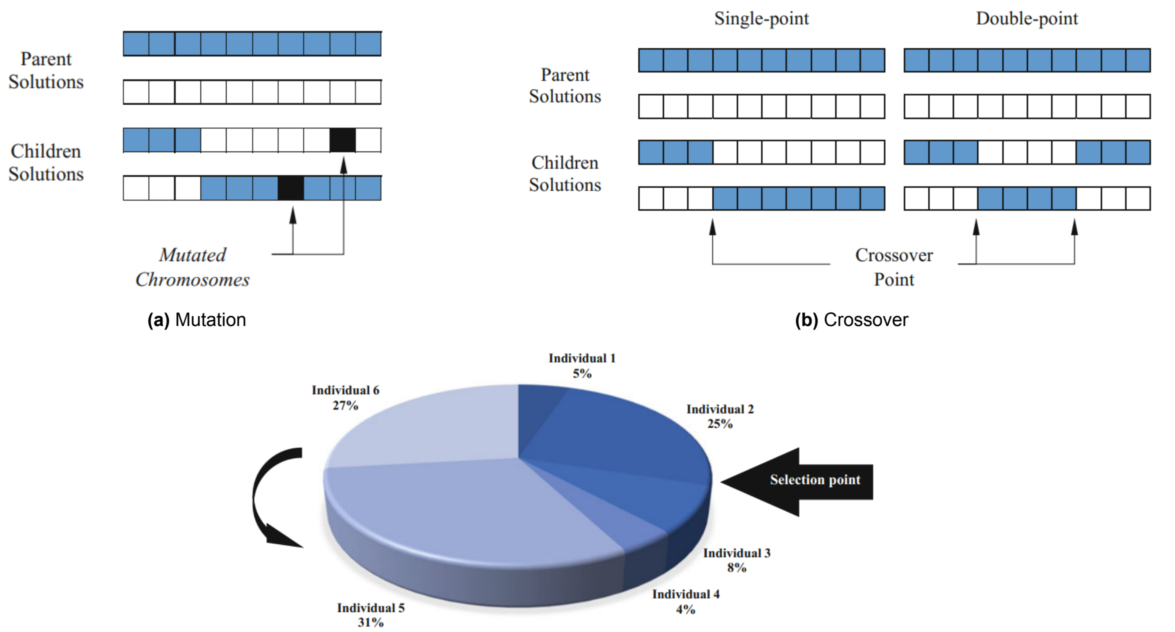
Figure 6.1: A flowchart of the tabu search technique [106].

When looking at tabu search research that relates to the SAP and the OPR, there are several examples. One of them has been discussed in Section 4.2.1, the research done by Chen et al. (2010) [35]. Other works include those of Xie et al. (2015), where they use a restricted neighbourhood tabu search to tackle the SAP [108]. They compared their algorithm to a genetic programming algorithm and tested twenty instances, for which the tabu search algorithm outperformed the genetic programming algorithm in all instances for both computational time and fitness, albeit with small margins. These margins became more significant if the problem size, or the number of items, increased.

Yet, the problem that is even more often tackled by the tabu search algorithm is the OPR problem. The work of Cortés et al. (2017) is a prime example of this, where they use a tabu search approach to solve the OPR for both large- and medium-sized FCs [109]. They use the general tabu search as well as two hybrid variations of the algorithm, to be able to compare them. The set-up that is used in their paper is very similar to set-ups of Picnic FCs, where a customer order cannot be split, there is a maximum storage capacity per storage location, the FC has a rectangular layout and there is a constraint set for the available inventory. For the experimental design the FC consisted of 200 storage locations along with 4 parallel aisles, where two-sided picking in an aisle is allowed. With regard to order picking time, the hybrid variation of the tabu search algorithm does outperform the general tabu search algorithm, albeit by no more than 7%. The performance of the general tabu search algorithm lies closer to the two benchmark algorithms that were also used, which consisted of a genetic algorithm and a simulated annealing approach. The outcome of this research underlines the fact that whilst tabu search is a very valid approach to this problem, there are still many adaptations that can be made to enhance the performance of the algorithm even further.

6.2. Genetic Algorithms

The genetic algorithm was inspired from the evolution theory as developed by Charles Darwin, as have been many so-called evolutionary computation methods, and was invented by John Holland in the 1960s [110]. It is a method that has been applied at Picnic for many uses as well and several master theses have also used this approach [111]. A genetic algorithm is a stochastic population-based algorithm, that uses operators like mutation, crossover and selection, which all have been inspired by nature [112]. A genetic algorithm starts with a random population, where each individual has a certain set of properties (chromosomes), which can be mutated. For each iteration that is done in the genetic algorithm, a new generation is determined. For each iteration the fitness of all individuals is evaluated, which is determined by the objective function of the MILP. These are then selected to continue with or not and a new generation is formed. The three main operators of mutation, crossover and selection are visualised in Figure 6.2. The operator mutation handles the alteration of the certain properties. This keeps diversity within the population due to certain randomness and hence avoids getting stuck in local optima. The mutation rate can be set by the creator of the algorithm. Crossover is the operator that handles the formation of a child solution from two parent solutions, how that also would happen in nature. There are multiple techniques that can be used for the crossover operations, two of which are shown in Figure 6.2b. Lastly, there is the operator of selection. This handles which individuals and their respective properties are selected for the future generations. Again, there are multiple techniques for this, one of which is shown in Figure 6.2c. This selection technique can be compared to a roulette wheel. The shares of individuals that have a higher fitness in terms of the objective function have a bigger share in this wheel and therefore a bigger chance of reproducing into the next generation [113].



(a) Mutation

(b) Crossover

(c) Selection

Figure 6.2: Visualisation of the main operators used in genetic algorithms [113].

With this in mind, a step-by-step approach to a genetic algorithm can be determined, which is found below.

Step 1: Initialise the population randomly.
Step 2: Evaluate the fitness of the current population to the objective function.
Step 3: Check if the stop condition is met, which can be set to a number of iterations or a level of fitness that is reached.
Step 4: If the stop condition is not met, select individuals for the new generation, perform crossover and mutation, and determine a new population. Repeat from step 2.
Step 5: If the stop condition is met, retrieve the final population and end the algorithm.

Genetic algorithms are often used in e-grocery research or in warehouse operations research in general. One of the application is to the OBP in FCs, as has been researched by Koch and Wäscher (2016) [114], to decrease the total length of all the pick rounds that are undertaken by order pickers. Each chromosome of an individual represents a batch of customer orders and since batches do not all have the same size, chromosomes are of different length as well. This entails that one individual is made out of several batches of customer orders. Selection, mutation and crossovers are applied, and it is checked whether all customer orders are included in the generation. This algorithm was tested in a warehouse, where experiments showed that the new developed algorithm decreased the total tour length by, on average, 3.15% compared to a previously developed genetic algorithm with a different approach (order based instead of group based). Next to that, computational time was also within reasonable limits. The authors draw the conclusion that a genetic algorithm is a verified good approach to the OBP, yet it remains very important to choose a genetic algorithm in such a manner that it is the most suitable for the problem, dependent on the characteristics of the problem.

Pan et al. (2015) also used the basis of a genetic algorithm for their approach to the SAP in a zone-picking FC, meaning that an order picker is responsible for a zone in which certain items are picked and then passes these picked items to an order picker in the next zone, who takes care of the item picking there [115]. With this, they try to determine which items need to be located into which zones. Here, an individual is a zone and the chromosomes represent certain items that are placed in that zone. After developing functions for the fitness, selection, crossover and mutation, a correction mechanism is also added to ensure a certain zone does not overflow in terms of maximum capacity. They test their algorithm in an FC with 60 racks and twenty units per rack, with different instances in terms of amount of picking zones. The objective is to minimise the time spent on order picking. The results show that certain bottleneck zones highly influence the performance, yet the algorithm performs better than storage policies and rules.

6.3. Ant Colony Optimisation

Just like a genetic algorithm, ant colony optimisation takes it inspiration from nature, from certain ant species. These ant species mark favorable paths with certain pheromones, such that other ants of the same colony know to follow the path as well. Ant colony optimisation belongs to the bigger group of swarm intelligence techniques, which are used more often to tackle optimisation problems [116]. Figure 6.3 shows the basic principle that ant colony optimisation is built upon.

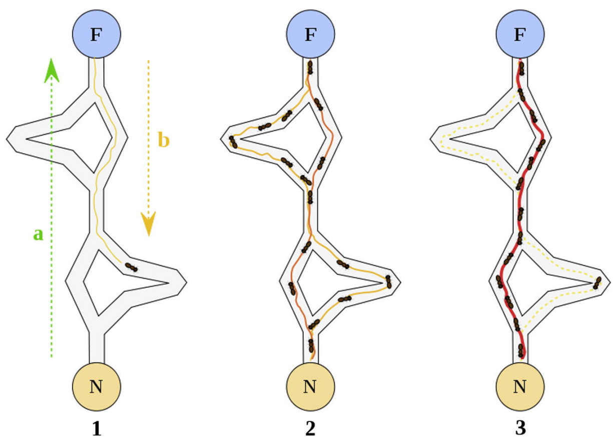


Figure 6.3: Ant colony optimisation and the shortest route that is chosen [117].

As can be seen from the figure, ant colony optimisation can be used especially for routing problems. Ant colony optimisation consists of three main phases, (i) construction, (ii) pheromone update, and (iii) daemon action, which is an optional phase. During the first phase, the construction phase, ants are created. Ants depict a set of variables and they move through different states of the problem by building paths, just like normal ants. By moving through the different states, the ants are construction solutions to the optimisation problem during the first phase. Once a solution is built, the ant will evaluate the solution and will deposit a pheromone trail for the other ants to follow. Pheromone updates, phase two, will take place over time and it decreases the pheromone left by the previous ants. This is to avoid getting stuck in local optima. Afterwards, daemon actions can be used to take actions that cannot be executed by a single ant, to check global information. As such, it can check the paths formed by all the ants and deposit additional pheromone on the most favourable one [118]. From this, the step-by-step approach to ant colony optimisation can be formulated as per the below.

Step 1: Initialise the problem with a set of ants.
Step 2: Let a random ant start construction and building a solution.
Step 3: Update the pheromone trail of the ant and add it the overall structure.
Step 4: Let the pheromone evaporate over time.
Step 5: (Optional) Perform a daemon action.
Step 6: Assess the stop condition in terms of the number of ants or a set termination criteria.
Step 7: If stop condition is not met, start with a new ant from step 2.
Step 8: If stop condition is met, evaluate best solution from the ant colony.

As mentioned, ant colony optimisation is very commonly used to solve the VRP and hence can be used to solve the OPR. This is exactly what the research conducted by Li et al. (2017) did, yet the authors combined the OBP with the OPR [119]. They adjusted the ant colony optimisation algorithm by adding a local search after a promising path was found. With an experiment of over 2000 SKUs and different order size instances (up to 10,000 orders) and the objective to minimise the order picking distance, their algorithm was evaluated. The algorithm improved, or reduced, the distance up to 80% compared to the traditional routing and batching of the FC. The authors mention that their algorithm shows very promising improvements and can handle large instances, however the actual problem keeps growing with continuing rise in demand and online sales. Therefore, existing meta-heuristics continuously need to be adapted and improved to still be able to solve the real life problems.

6.4. Large Neighbourhood Search

Large neighbourhood searches are commonly used in routing and scheduling problems. The large neighbourhood search was first introduced by Paul Shaw in 1998 [120]. Like the other methods, a large neighbourhood search starts from an initial solution which is gradually improved. This happens by continuously destroying and repairing the solution until a stop condition is met [121]. Before this, a (large) neighbourhood needs to be defined, in which the optimal solution needs to be found through the destruction and repair processes. Within the destruction part, or the relaxation part, there usually is a stochastic element, such that different parts of the solution are taken down. After the solution is rebuild, or re-optimised, it needs to be determined if the new solution will be accepted or not, usually dependent on if the solution is improving or not. One iteration of the destruction and repair of certain positions is corresponding to the examination of one neighbourhood move. There are multiple options to choose how the positions are destroyed and repaired, hence making the algorithm very flexible [122]. Figure 6.4 shows an example of the destruction and repair process. The process of a large neighbourhood search can be described with the steps as outlined below.

Step 1: Start with an initial solution.
Step 2: Destroy part of the solution by removing certain positions, based on a previously determined method.

Step 3: Repair paths to the just removed positions, based on a previously determine method.

Step 4: Evaluate the new solution and determine if it is better than the current solution and the best solution that is saved.

Step 5: Assess the stop condition in previously set terms of time and/or iterations.

Step 6: If better than the current solution, keep the solution and start with this as a new solution from step 2. Save the solution as the best solution if that is the case.

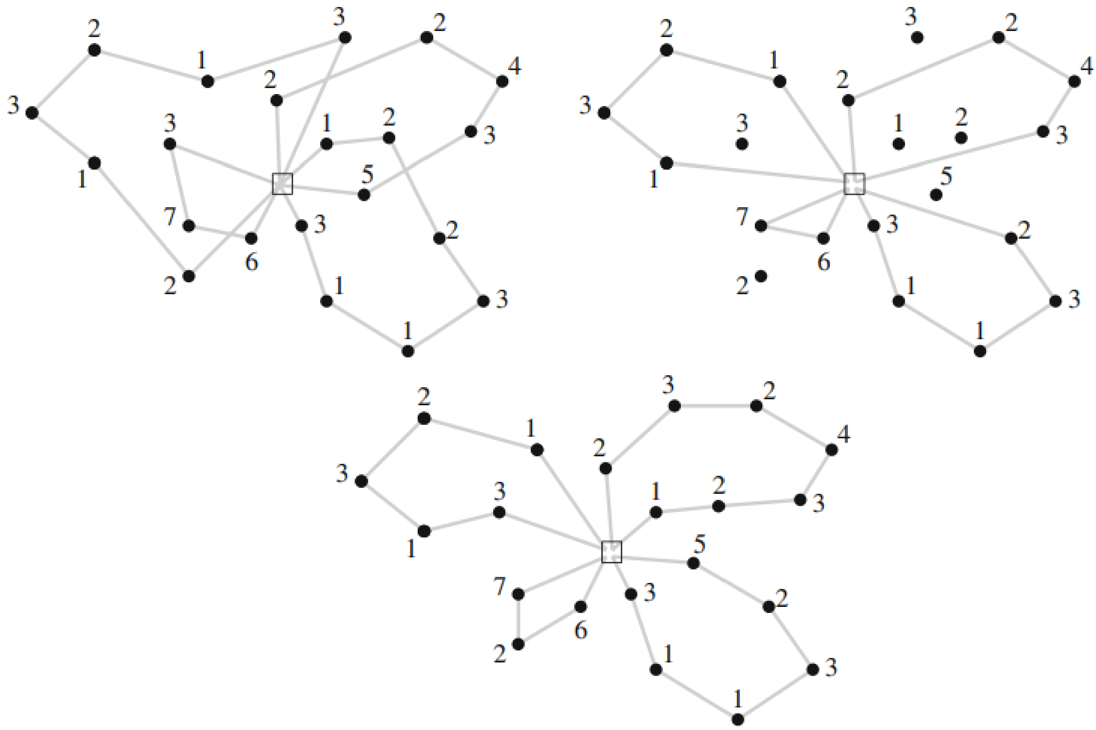


Figure 6.4: An example of destruction and repair, where the top left figure shows an initial solution, the top right the solution after a destroy operation and the bottom figure the solution after a repair operation [121].

One version of the large neighbourhood search is the adaptive large neighbourhood search, which was proposed by Ropke and Pisinger (2006) for a version of the LMDP [123]. The algorithm was tested against more than 350 known instances from literature, for which it showed improvement for more than 50% of the problems. Different from the normal large neighbourhood search, the adaptive large neighbourhood search allows for the use of multiple destroy and repair methods, instead of only one. Via a weight distribution it is determined how often the method is attempted during the neighbourhood search. These weights are adjusted or adapted throughout the search, such that it makes a very dynamic algorithm. Often, it is also needed to determine a trade-off between the time consumption of a certain method and the optimality of the solution, to avoid the algorithm from only favouring complex repair methods.

For problems within e-commerce, (adaptive) large neighbourhood searches are often chosen as meta-heuristic. One of which was the research done by Chabot et al. (2017), which shows that the adaptive large neighbourhood search is an excellent candidate to solve OPR with the objective to reduce order picking distance [124]. When comparison is made between their adaptive large neighbourhood search and classic heuristics such as the S-shape or largest gap method, as described in Section 4.2.3, the adaptive large neighbourhood search consistently outperforms the heuristic methods.

Another example is the paper of Kuhn et al. (2021), where they use an addition to the adaptive large neighbourhood search for the integrated OBP, OPR and VRP of delivery to reduce tardiness

[125]. When testing this algorithm, it performs better than the traditional adaptive large neighbourhood search in the objective of decreasing tardiness, which is the same as decreasing time or improving efficiency. Hence, adjusting the adaptive large neighbourhood search to fit to a problem can still show significant improvements and are worth the investigation, especially in e-groceries.

References

[1] Sanne Schelfaut. *Groei online boodschappen doen voorbij: ouderen zelf op pad*. July 2021.

[2] *De onlineboodschappentaart is veel groter, het stuk van AH kleiner*. Feb. 2021.

[3] Arkadiusz Kawa. "FULFILLMENT SERVICE IN E-COMMERCE LOGISTICS". In: *Logforum* 13.4 (Dec. 2017), pp. 429–438. ISSN: 18952038. DOI: 10.17270/J.LOG.2017.4.4. URL: http://www.logforum.net/volume13/issue4/abstract-4.html.

[4] C.K.M. Lee et al. "Design and application of Internet of things-based warehouse management system for smart logistics". In: *International Journal of Production Research* 56.8 (Apr. 2018), pp. 2753–2768. ISSN: 0020-7543. DOI: 10.1080/00207543.2017.1394592. URL: https://www.tandfonline.com/doi/full/10.1080/00207543.2017.1394592.

[5] René de Koster, Tho Le-Duc, and Kees Jan Roodbergen. "Design and control of warehouse order picking: A literature review". In: *European Journal of Operational Research* 182.2 (Oct. 2007), pp. 481–501. ISSN: 03772217. DOI: 10.1016/j.ejor.2006.07.009.

[6] H. Donald Ratliff and Arnon S. Rosenthal. "Order-Picking in a Rectangular Warehouse: A Solvable Case of the Traveling Salesman Problem". In: *Operations Research* 31.3 (June 1983), pp. 507–521. ISSN: 0030-364X. DOI: 10.1287/opre.31.3.507.

[7] Juan José Rojas Reyes, Elyn Lizeth Solano-Charris, and Jairo Rafael Montoya-Torres. *The storage location assignment problem: A literature review*. Apr. 2019. DOI: 10.5267/j.ijiec.2018.8.001.

[8] K.K.H. Ng et al. "Review on meta-heuristics approaches for airside operation research". In: *Applied Soft Computing* 66 (May 2018), pp. 104–133. ISSN: 15684946. DOI: 10.1016/j.asoc.2018.02.013.

[9] Wei Jiang et al. "Assignment of duplicate storage locations in distribution centres to minimise walking distance in order picking". In: *International Journal of Production Research* 59.15 (2021), pp. 4457–4471. ISSN: 1366588X. DOI: 10.1080/00207543.2020.1766714.

[10] Riccardo Accorsi, Riccardo Manzini, and Fausto Maranesi. "A decision-support system for the design and management of warehousing systems". In: *Computers in Industry* 65.1 (Jan. 2014), pp. 175–186. ISSN: 01663615. DOI: 10.1016/j.compind.2013.08.007.

[11] Rachel Bellamy et al. *Exploring A New Paradigm for E-Groceries*. Tech. rep. 2000. URL: https://www.researchgate.net/publication/228402561.

[12] Nielsen Global E-commerce. *The future of grocery*. Tech. rep. The Nielsen Company, 2015. URL: https://www.nielsen.com/wp-content/uploads/sites/3/2019/04/nielsen-global-e-commerce-new-retail-report-april-2015.pdf.

[13] Marcia Mkansi, Chukuakadibia Eresia-Eke, and Oyetola Emmanuel-Ebikake. "E-grocery challenges and remedies: Global market leaders perspective". In: *Cogent Business & Management* 5.1 (Jan. 2018), p. 1459338. ISSN: 2331-1975. DOI: 10.1080/23311975.2018.1459338.

[14] Charles Dennis, Tino Fenech, and Bill Merrilees. "E-retailing". In: (2004).

[15] Mark B. Kolesar and R. Wayne Galbraith. "A services□marketing perspective on e□retailing: implications for e□retailers and directions for further research". In: *Internet Research* 10.5 (Dec. 2000), pp. 424–438. ISSN: 1066-2243. DOI: 10.1108/10662240010349444.

[16] Eleonora Pantano et al. *Internet Retailing and Future Perspectives*. Routledge, Dec. 2016. ISBN: 9781317378778. DOI: 10.4324/9781315674308.

[17] Alexander Hermann Hübner, Heinrich Kuhn, and Johannes Wollenburg. "Last mile fulfilment and distribution in omni-channel grocery retailing: a strategic planning framework". In: *International Journal of Retail & Distribution Management* 44.3 (Mar. 2016). ISSN: 0959-0552. DOI: 10.1108/IJRDM-11-2014-0154.

[18] Frank Wiengarten et al. “Assessing the value creation process of e□business along the supply chain”. In: *Supply Chain Management: An International Journal* 16.4 (June 2011), pp. 207–219. ISSN: 1359-8546. DOI: 10.1108/13598541111139035.

[19] Coralis Berggren and Sarah Wikstrom. “Barriers Online: Exploring Consumers’ Resistance to E-groceries”. PhD thesis. Jönköping University, 2018.

[20] Valerio Gatta et al. “E-groceries and urban freight: Investigating purchasing habits, peer influence and behaviour change via a discrete choice/agent-based modelling approach”. In: *Transportation Research Procedia* 46 (Jan. 2020), pp. 133–140. ISSN: 2352-1465. DOI: 10.1016/J.TRPRO.2020.03.173.

[21] Stephanie Chevalier. *Share of supermarket revenue that is generated through online grocery shopping from October 2015 to June 2020*. Feb. 2022.

[22] Juan Carlos Martín, Francesca Pagliara, and Concepción Román. “The research topics on e-grocery: Trends and existing gaps”. In: *Sustainability (Switzerland)* 11.2 (Jan. 2019). ISSN: 20711050. DOI: 10.3390/su11020321.

[23] Yeming Gong and René B. M. de Koster. “A review on stochastic models and analysis of warehouse operations”. In: *Logistics Research* 3.4 (Oct. 2011), pp. 191–205. ISSN: 1865-035X. DOI: 10.1007/s12159-011-0057-6.

[24] Jinxiang Gu, Marc Goetschalckx, and Leon F. McGinnis. “Research on warehouse design and performance evaluation: A comprehensive review”. In: *European Journal of Operational Research* 203.3 (June 2010), pp. 539–549. ISSN: 0377-2217. DOI: 10.1016/J.EJOR.2009.07.031.

[25] Gilles Cormier and Eldon A. Gunn. “A review of warehouse models”. In: *European Journal of Operational Research* 58.1 (Apr. 1992), pp. 3–13. ISSN: 0377-2217. DOI: 10.1016/0377-2217(92)90231-W.

[26] Edward H. Frazelle. “Small parts order picking: equipment and strategy”. In: *Warehousing Education and Research Council* 88 (1988).

[27] Donald Ervin Knuth. “Postscript about NP-hard problems”. In: *ACM SIGACT News* 6 (1974), pp. 15–16.

[28] Daniel Pierre Bovet, Pierluigi Crescenzi, and D. Bovet. *Introduction to the Theory of Complexity*. Vol. 7. Prentice Hall London, 1994.

[29] T. Le-Duc * and R.(M.)B.M. De Koster. “Travel distance estimation and storage zone optimization in a 2-block class-based storage strategy warehouse”. In: *International Journal of Production Research* 43.17 (Sept. 2005), pp. 3561–3581. ISSN: 0020-7543. DOI: 10.1080/00207540500142894.

[30] *Solving mixed integer programming problems (MIP)*. 2021.

[31] *Mixed-Integer Programming (MIP) – A Primer on the Basics*. 2022.

[32] Teofilo F. Gonzalez. *Handbook of Approximation Algorithms and Metaheuristics*. Ed. by Teofilo F. Gonzalez. Chapman and Hall/CRC, May 2007. ISBN: 9780429143793. DOI: 10.1201/9781420010749.

[33] Nasser A. El-Sherbeny. “Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods”. In: *Journal of King Saud University - Science* 22.3 (July 2010), pp. 123–131. ISSN: 1018-3647. DOI: 10.1016/J.JKSUS.2010.03.002.

[34] Daria Battini et al. “Order picking system design: the storage assignment and travel distance estimation (SA&TDE) joint method”. In: *International Journal of Production Research* 53.4 (Feb. 2015), pp. 1077–1093. ISSN: 0020-7543. DOI: 10.1080/00207543.2014.944282.

[35] Lu Chen, Andr Langevin, and Diane Riopel. “A tabu search algorithm for the relocation problem in a warehousing system”. In: *International Journal of Production Economics* 129.1 (Jan. 2011), pp. 147–156. ISSN: 0925-5273. DOI: 10.1016/J.IJPE.2010.09.012.

[36] Fred Glover and Manuel Laguna. “Tabu Search”. In: *Handbook of Combinatorial Optimization*. Boston, MA: Springer US, 1998, pp. 2093–2229. DOI: 10.1007/978-1-4613-0303-9{_}33.

[37] Marcele Elisa Fontana and Cristiano Alexandre Virgínio Cavalcante. “Using the Efficient Frontier to Obtain the Best Solution for the Storage Location Assignment Problem”. In: *Mathematical Problems in Engineering* 2014 (2014), pp. 1–10. ISSN: 1024-123X. DOI: 10.1155/2014/745196.

[38] Abou Bakr Belkaid et al. *Assessment of Performance of a Class-Based Storage in a Flow-Rack AS/RS Textbook writing View project TUBITAK 214M613-Energy Efficient Autonomous Vehicle Based Warehouse System Design View project Nihad MEGHELLI GAOUAR Assessment of Performance of a Class-Based Storage in a Flow-Rack AS/RS*. Tech. rep. 2010. URL: https://www.researchgate.net/publication/260300977.

[39] Ralf Elbert and Jan Philipp Müller. “Manoeuvres in manual person-to-parts order picking systems: Is there a significant influence on travel time and metabolic rates?” In: *IFAC-PapersOnLine* 50.1 (July 2017), pp. 6894–6899. ISSN: 2405-8963. DOI: 10.1016/J.IFACOL.2017.08.1213.

[40] Stanley Frederick W.T. Lim, Xin Jin, and Jagjit Singh Srai. “Consumer-driven e-commerce”. In: *International Journal of Physical Distribution & Logistics Management* 48.3 (Mar. 2018), pp. 308–332. ISSN: 0960-0035. DOI: 10.1108/IJPDLM-02-2017-0081.

[41] Riccardo Mangiaracina et al. “Innovative solutions to increase last-mile delivery efficiency in B2C e-commerce: a literature review”. In: *International Journal of Physical Distribution & Logistics Management* 49.9 (Nov. 2019), pp. 901–920. ISSN: 0960-0035. DOI: 10.1108/IJPDLM-02-2019-0048.

[42] S. Geetha, G. Poonthalir, and P.T. Vanathi. “Nested particle swarm optimisation for multi-depot vehicle routing problem”. In: *International Journal of Operational Research* 16.3 (2013), p. 329. ISSN: 1745-7645. DOI: 10.1504/IJOR.2013.052336.

[43] László Kovács, Anita Agárdi, and Tamás Bányai. “Fitness Landscape Analysis and Edge Weighting-Based Optimization of Vehicle Routing Problems”. In: *Processes* 8.11 (Oct. 2020), p. 1363. ISSN: 2227-9717. DOI: 10.3390/pr8111363.

[44] Max Leyerer et al. “Shortening the Last Mile in Urban Areas: Optimizing a Smart Logistics Concept for E-Grocery Operations”. In: *Smart Cities* 3.3 (July 2020), pp. 585–603. ISSN: 2624-6511. DOI: 10.3390/smartcities3030031.

[45] Martin Waitz, Andreas Mild, and Christian Fikar. “A Decision Support System for Efficient Last-Mile Distribution of Fresh Fruits and Vegetables as Part of E-Grocery Operations”. In: *51st Hawaii International Conference on System Sciences* (2018), pp. 1259–1267. URL: http://hdl.handle.net/10125/50042.

[46] Yuan Wang et al. “Towards enhancing the last-mile delivery: An effective crowd-tasking model with scalable solutions”. In: *Transportation Research Part E: Logistics and Transportation Review* 93 (Sept. 2016), pp. 279–293. ISSN: 1366-5545. DOI: 10.1016/J.TRE.2016.06.002.

[47] Kevin Dorling et al. “Vehicle Routing Problems for Drone Delivery”. In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 47.1 (Jan. 2017), pp. 70–85. ISSN: 2168-2216. DOI: 10.1109/TSMC.2016.2582745.

[48] Nils Boysen, René de Koster, and Felix Weidinger. *Warehousing in the e-commerce era: A survey*. Sept. 2019. DOI: 10.1016/j.ejor.2018.08.023.

[49] Masa Slabinac. “INNOVATIVE SOLUTIONS FOR A “LAST-MILE” DELIVERY – A EUROPEAN EXPERIENCE”. In: *15th international scientific conference Business Logistics in Modern Management* (2015), pp. 111–129.

[50] James A Tompkins et al. *Facilities planning*. John Wiley & Sons, 2010.

[51] Tho Le Duc and René de Koster. “Travel Distance Estimation in Single-block ABC- Storage Strategy Warehouses”. In: 2005, pp. 185–200. DOI: 10.1007/978-3-642-17020-1{_}10.

[52] Rajesh Matai, Murari Lal Mittal, and Surya Prakash Singh. *Traveling salesman problem: an overview of applications, formulations, and solution approaches*. Vol. 1. InTech, 2010. ISBN: 9789533074269.

[53] Adam N. Letchford, Saeideh D. Nasiri, and Dirk Oliver Theis. “Compact formulations of the Steiner Traveling Salesman Problem and related problems”. In: *European Journal of Operational Research* 228.1 (July 2013), pp. 83–92. ISSN: 0377-2217. DOI: 10.1016/J.EJOR.2013.01.044.

[54] RENÉ DE KOSTER and EDO VAN DER POORT. “Routing orderpickers in a warehouse: a comparison between optimal and heuristic solutions”. In: *IIE Transactions* 30.5 (1999), pp. 469–480. ISSN: 0740817X. DOI: 10.1023/A:1007599307171.

[55] NOUD GADEMANN and STEEF VELDE. “Order batching to minimize total travel time in a parallel-aisle warehouse”. In: *IIE Transactions* 37.1 (Jan. 2005), pp. 63–75. ISSN: 0740-817X. DOI: 10.1080/07408170590516917.

[56] RANDOLPH W. HALL. “DISTANCE APPROXIMATIONS FOR ROUTING MANUAL PICKERS IN A WAREHOUSE”. In: *IIE Transactions* 25.4 (July 1993), pp. 76–87. ISSN: 0740-817X. DOI: 10.1080/07408179308964306.

[57] Fangyu Chen, Gangyan Xu, and Yongchang Wei. “Heuristic routing methods in multiple-block warehouses with ultra-narrow aisles and access restriction”. In: *International Journal of Production Research* 57.1 (Jan. 2019), pp. 228–249. ISSN: 0020-7543. DOI: 10.1080/00207543.2018.1473657.

[58] Charles G. Petersen. “An evaluation of order picking routeing policies”. In: *International Journal of Operations & Production Management* 17.11 (Nov. 1997), pp. 1098–1111. ISSN: 0144-3577. DOI: 10.1108/01443579710177860.

[59] Namrata Shetty, Bhawesh Sah, and Sung Hoon Chung. “Route optimization for warehouse order picking operations via vehicle routing and simulation”. In: *SN Applied Sciences* 2.2 (Feb. 2020). ISSN: 25233971. DOI: 10.1007/s42452-020-2076-x.

[60] M. Klodawski et al. “Simulation analysis of order picking efficiency with congestion situations”. In: *International Journal of Simulation Modelling* 17.3 (Sept. 2018), pp. 431–443. ISSN: 19968566. DOI: 10.2507/IJSIMM17(3)438.

[61] Ralf M. Elbert et al. “The effects of human behavior on the efficiency of routing policies in order picking: The case of route deviations”. In: *Computers & Industrial Engineering* 111 (Sept. 2017), pp. 537–551. ISSN: 0360-8352. DOI: 10.1016/J.CIE.2016.11.033.

[62] David R. Gibson and Gunter P. Sharp. “Order batching procedures”. In: *European Journal of Operational Research* 58.1 (Apr. 1992), pp. 57–67. ISSN: 0377-2217. DOI: 10.1016/0377-2217(92)90235-2.

[63] James A. Chisman. “The clustered traveling salesman problem”. In: *Computers & Operations Research* 2.2 (Sept. 1975), pp. 115–119. ISSN: 0305-0548. DOI: 10.1016/0305-0548(75)90015-5.

[64] Y. A. Bozer and J. W. Kile. “Order batching in walk-and-pick order picking systems”. In: *International Journal of Production Research* 46.7 (Apr. 2008), pp. 1887–1909. ISSN: 0020-7543. DOI: 10.1080/00207540600920850.

[65] Robert A. Ruben and F. Robert Jacobs. “Batch Construction Heuristics and Storage Assignment Strategies for Walk/Ride and Pick Systems”. In: *Management Science* 45.4 (Apr. 1999), pp. 575–596. ISSN: 0025-1909. DOI: 10.1287/mnsc.45.4.575.

[66] Çağla Cergibozan and A. Serdar Tasan. “Order batching operations: an overview of classification, solution techniques, and future research”. In: *Journal of Intelligent Manufacturing* 30.1 (Jan. 2019), pp. 335–349. ISSN: 0956-5515. DOI: 10.1007/s10845-016-1248-4.

[67] Sebastian Henn et al. “Metaheuristics for the Order Batching Problem in Manual Order Picking Systems”. In: *Business Research* 3 (2010), pp. 82–105.

[68] Sebastian Henn and Gerhard Wäscher. “Tabu search heuristics for the order batching problem in manual order picking systems”. In: *European Journal of Operational Research* 222.3 (Nov. 2012), pp. 484–494. ISSN: 0377-2217. DOI: 10.1016/J.EJOR.2012.05.049.

[69] Çağla Cergibozan and A. Serdar Tasan. “Genetic algorithm based approaches to solve the order batching problem and a case study in a distribution center”. In: *Journal of Intelligent Manufacturing* 33.1 (Jan. 2022), pp. 137–149. ISSN: 0956-5515. DOI: 10.1007/s10845-020-01653-3.

[70] P Haris Kumar and R Mageshvaran. “Methods and solvers used for solving mixed integer linear programming and mixed nonlinear programming problems: A review”. In: *International Journal of Scientific and Technology Research* 9.1 (2020), pp. 1872–1882.

[71] E. L. Lawler and D. E. Wood. “Branch-and-Bound Methods: A Survey”. In: *Operations Research* 14.4 (Aug. 1966), pp. 699–719. ISSN: 0030-364X. DOI: 10.1287/opre.14.4.699.

[72] Stefan Poikonen, Bruce Golden, and Edward A. Wasil. “A Branch-and-Bound Approach to the Traveling Salesman Problem with a Drone”. In: *INFORMS Journal on Computing* 31.2 (Apr. 2019), pp. 335–346. ISSN: 1091-9856. DOI: 10.1287/ijoc.2018.0826.

[73] Gerald J Lieberman and Frederick S Hillier. *Introduction to operations research*. Vol. 8. New York: McGraw-Hill New York, NY, USA, 2005.

[74] He He, Hal Daume III, and Jason M Eisner. “Learning to search in branch and bound algorithms”. In: *Advances in neural information processing systems* 27 (2014).

[75] Basheer M. Khumawala. “An Efficient Branch and Bound Algorithm for the Warehouse Location Problem”. In: *Management Science* 18.12 (Aug. 1972), B-718-B–731. ISSN: 0025-1909. DOI: 10.1287/mnsc.18.12.B718.

[76] Sandra Hahn and Andre Scholz. “Order Picking in Narrow-Aisle Warehouses, A Fast Approach to Minimize Waiting Times”. In: *Working Paper Series* 1 (2017).

[77] Mohamed Ali Rakrouki, Talel Ladhari, and Vincent T’Kindt. “Coupling Genetic Local Search and Recovering Beam Search algorithms for minimizing the total completion time in the single machine scheduling problem subject to release dates”. In: *Computers & Operations Research* 39.6 (June 2012), pp. 1257–1264. ISSN: 0305-0548. DOI: 10.1016/J.COR.2010.12.007.

[78] Venkata Reddy Muppani (Muppant) and Gajendra Kumar Adil. “A branch and bound algorithm for class based storage location assignment”. In: *European Journal of Operational Research* 189.2 (Sept. 2008), pp. 492–507. ISSN: 03772217. DOI: 10.1016/j.ejor.2007.05.050.

[79] Kashi N. Singh and Dirk L. Van Oudheusden. “A branch and bound algorithm for the traveling purchaser problem”. In: *European Journal of Operational Research* 97.3 (Mar. 1997), pp. 571–579. ISSN: 0377-2217. DOI: 10.1016/S0377-2217(96)00313-X.

[80] Cynthia Barnhart et al. “A Branch-and-Price Algorithm for the Generalized Assignment Problem”. In: *Operations Research* 45.6 (Dec. 1997), pp. 831–841. ISSN: 0030-364X. DOI: 10.1287/opre.45.6.831.

[81] François Vanderbeck and Martin W.P. Savelsbergh. “A generic view of Dantzig–Wolfe decomposition in mixed integer programming”. In: *Operations Research Letters* 34.3 (May 2006), pp. 296–306. ISSN: 0167-6377. DOI: 10.1016/J.ORL.2005.05.009.

[82] Dominique Feillet. “A tutorial on column generation and branch-and-price for vehicle routing problems”. In: *4OR* 8.4 (Dec. 2010), pp. 407–424. ISSN: 1619-4500. DOI: 10.1007/s10288-010-0130-z.

[83] Sattrawut Ponboon, Ali Gul Qureshi, and Eiichi Taniguchi. “Branch-and-price algorithm for the location-routing problem with time windows”. In: *Transportation Research Part E: Logistics and Transportation Review* 86 (Feb. 2016), pp. 1–19. ISSN: 1366-5545. DOI: 10.1016/J.TRE.2015.12.003.

[84] Alberto Ceselli and Giovanni Righini. “A Branch-and-Price Algorithm for the Multilevel Generalized Assignment Problem”. In: *Operations Research* 54.6 (Dec. 2006), pp. 1172–1184. ISSN: 0030-364X. DOI: 10.1287/opre.1060.0323.

[85] M. Firat, D. Briskorn, and A. Laugier. “A Branch-and-Price algorithm for stable workforce assignments with hierarchical skills”. In: *European Journal of Operational Research* 251.2 (June 2016), pp. 676–685. ISSN: 0377-2217. DOI: 10.1016/J.EJOR.2015.11.039.

[86] Xu Wang et al. “A Branch and Price Algorithm for Crane Assignment and Scheduling in Slab Yard”. In: *IEEE Transactions on Automation Science and Engineering* 18.3 (July 2021), pp. 1122–1133. ISSN: 1545-5955. DOI: 10.1109/TASE.2020.2996227.

[87] Nabila Azi, Michel Gendreau, and Jean Yves Potvin. “An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles”. In: *European Journal of Operational Research* 202.3 (May 2010), pp. 756–763. ISSN: 0377-2217. DOI: 10.1016/J.EJOR.2009.06.034.

[88] Jacques Desrosiers and Marco E. Lübbecke. "Branch-Price-and-Cut Algorithms". In: *Wiley Encyclopedia of Operations Research and Management Science*. Hoboken, NJ, USA: John Wiley & Sons, Inc., Jan. 2011. DOI: 10.1002/9780470400531.eorms0118.

[89] Claudia Archetti, Nicola Bianchessi, and M. Grazia Speranza. "A branch-price-and-cut algorithm for the commodity constrained split delivery vehicle routing problem". In: *Computers & Operations Research* 64 (Dec. 2015), pp. 1–10. ISSN: 0305-0548. DOI: 10.1016/J.COR.2015.04.023.

[90] Qie He, Stefan Irnich, and Yongjia Song. "Branch-and-Cut-and-Price for the Vehicle Routing Problem with Time Windows and Convex Node Costs". In: *Transportation Science* 53.5 (Sept. 2019), pp. 1409–1426. ISSN: 0041-1655. DOI: 10.1287/trsc.2019.0891.

[91] Roni Stern. "Multi-Agent Path Finding – An Overview". In: 2019, pp. 96–115. DOI: 10.1007/978-3-030-33274-7{_}6.

[92] Edward Lam et al. "Branch-and-cut-and-price for multi-agent path finding". In: *Computers & Operations Research* 144 (Aug. 2022), p. 105809. ISSN: 0305-0548. DOI: 10.1016/J.COR.2022.105809. URL: https://linkinghub.elsevier.com/retrieve/pii/S0305054822000946.

[93] L.R. Ford and D.R. Fulkerson. "A suggested computation for maximal multicommodity network flows". In: *Management Science* 5 (1958), pp. 97–101.

[94] Wilbert E. Wilhelm. "A Technical Review of Column Generation in Integer Programming". In: *Optimization and Engineering* 2.2 (2001), pp. 159–200. ISSN: 13894420. DOI: 10.1023/A:1013141227104.

[95] Duygu Taş. "Electric vehicle routing with flexible time windows: a column generation solution approach". In: *Transportation Letters* 13.2 (Feb. 2021), pp. 97–103. ISSN: 1942-7867. DOI: 10.1080/19427867.2020.1711581.

[96] İbrahim Muter, Ş. İlker Birbil, and Kerem Bülbül. "Simultaneous column-and-row generation for large-scale linear programs with column-dependent-rows". In: *Mathematical Programming* 142.1-2 (Dec. 2013), pp. 47–82. ISSN: 0025-5610. DOI: 10.1007/s10107-012-0561-8.

[97] Ehsan Ardjmand et al. "A hybrid artificial neural network, genetic algorithm and column generation heuristic for minimizing makespan in manual order picking operations". In: *Expert Systems with Applications* 159 (Nov. 2020), p. 113566. ISSN: 0957-4174. DOI: 10.1016/J.ESWA.2020.113566.

[98] Luigi Moccia et al. "A column generation heuristic for a dynamic generalized assignment problem". In: *Computers & Operations Research* 36.9 (Sept. 2009), pp. 2670–2681. ISSN: 0305-0548. DOI: 10.1016/J.COR.2008.11.022.

[99] İbrahim Muter and Temel Öncan. "Order batching and picker scheduling in warehouse order picking". In: *IISE Transactions* (June 2021), pp. 1–13. ISSN: 2472-5854. DOI: 10.1080/24725854.2021.1925178.

[100] Matthias Templ and B Meindl. *Analysis of commercial and free and open source solvers for linear optimization problems*. Tech. rep. Vienna: Institut f. Statistik u. Wahrscheinlichkeitstheorie, 2012. URL: http://www.statistik.tuwien.ac.at.

[101] Felix Weidinger, Nils Boysen, and Dirk Briskorn. "Storage Assignment with Rack-Moving Mobile Robots in KIVA Warehouses". In: *Transportation Science* 52.6 (Dec. 2018), pp. 1479–1495. ISSN: 0041-1655. DOI: 10.1287/trsc.2018.0826.

[102] Chen Yang Cheng et al. "Using a hybrid approach based on the particle swarm optimization and ant colony optimization to solve a joint order batching and picker routing problem". In: *International Journal of Production Economics* 170 (Dec. 2015), pp. 805–814. ISSN: 0925-5273. DOI: 10.1016/J.IJPE.2015.03.021.

[103] Fred Glover. "Future paths for integer programming and links to artificial intelligence". In: *Computers & Operations Research* 13.5 (Jan. 1986), pp. 533–549. ISSN: 0305-0548. DOI: 10.1016/0305-0548(86)90048-1.

[104] *Tabu*. 2022.

[105] A. Stefek. “Benchmarking of heuristic optimization methods”. In: *14th International Conference Mechatronika*. IEEE, June 2011, pp. 68–71. ISBN: 978-80-8075-477-8. DOI: 10.1109/MECHATRON.2011.5961068.

[106] Fred Glover. “Tabu Search: A Tutorial”. In: *Interfaces* 20.4 (Aug. 1990), pp. 74–94. ISSN: 0092-2102. DOI: 10.1287/inte.20.4.74.

[107] Yu Lin, Zheyong Bian, and Xiang Liu. “Developing a dynamic neighborhood structure for an adaptive hybrid simulated annealing – tabu search algorithm to solve the symmetrical traveling salesman problem”. In: *Applied Soft Computing* 49 (Dec. 2016), pp. 937–952. ISSN: 1568-4946. DOI: 10.1016/J.ASOC.2016.08.036.

[108] Jing Xie et al. “A restricted neighbourhood Tabu Search for Storage Location Assignment Problem”. In: *2015 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, May 2015, pp. 2805–2812. ISBN: 978-1-4799-7492-4. DOI: 10.1109/CEC.2015.7257237.

[109] Pablo Cortés et al. “A tabu search approach to solving the picking routing problem for large- and medium-size distribution centres considering the availability of inventory and K heterogeneous material handling equipment”. In: *Applied Soft Computing* 53 (Apr. 2017), pp. 61–73. ISSN: 1568-4946. DOI: 10.1016/J.ASOC.2016.12.026.

[110] Rubén Aguilar-Rivera, Manuel Valenzuela-Rendón, and J. J. Rodríguez-Ortiz. “Genetic algorithms and Darwinian approaches in financial applications: A survey”. In: *Expert Systems with Applications* 42.21 (Nov. 2015), pp. 7684–7697. ISSN: 0957-4174. DOI: 10.1016/J.ESWA.2015.06.001.

[111] Geert Konijnendijk. *It’s in our DNA: Solving logistics problems using Genetic Algorithms*. Jan. 2021.

[112] Melanie Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[113] Seyedali Mirjalili. “Genetic Algorithm”. In: 2019, pp. 43–55. DOI: 10.1007/978-3-319-93025-1{_}4.

[114] Sören Koch and Gerhard Wäscher. “A grouping genetic algorithm for the Order Batching Problem in distribution warehouses”. In: *Journal of Business Economics* 86.1-2 (Jan. 2016), pp. 131–153. ISSN: 0044-2372. DOI: 10.1007/s11573-015-0789-x.

[115] Jason Chao Hsien Pan et al. “A storage assignment heuristic method based on genetic algorithm for a pick-and-pass warehousing system”. In: *Computers & Industrial Engineering* 81 (Mar. 2015), pp. 1–13. ISSN: 0360-8352. DOI: 10.1016/J.CIE.2014.12.010.

[116] Marco Dorigo, Mauro Birattari, and Thomas Stutzle. “Ant colony optimization”. In: *IEEE Computational Intelligence Magazine* 1.4 (Nov. 2006), pp. 28–39. ISSN: 1556-603X. DOI: 10.1109/MCI.2006.329691.

[117] M. Duran Toksari. “A hybrid algorithm of Ant Colony Optimization (ACO) and Iterated Local Search (ILS) for estimating electricity domestic consumption: Case of Turkey”. In: *International Journal of Electrical Power & Energy Systems* 78 (June 2016), pp. 776–782. ISSN: 0142-0615. DOI: 10.1016/J.IJEPES.2015.12.032.

[118] Marco Dorigo and Thomas Stützle. “The Ant Colony Optimization Metaheuristic: Algorithms, Applications, and Advances”. In: 2003, pp. 250–285. DOI: 10.1007/0-306-48056-5{_}9.

[119] Jianbin Li, Rihuan Huang, and James B. Dai. “Joint optimisation of order batching and picker routing in the online retailer’s warehouse in China”. In: *International Journal of Production Research* 55.2 (Jan. 2017), pp. 447–461. ISSN: 0020-7543. DOI: 10.1080/00207543.2016.1187313.

[120] David Pisinger and Stefan Ropke. “Large Neighborhood Search”. In: 2019, pp. 99–127. DOI: 10.1007/978-3-319-91086-4{_}4.

[121] David Pisinger and Stefan Ropke. “Large Neighborhood Search”. In: 2010, pp. 399–419. DOI: 10.1007/978-1-4419-1665-5{_}13.

[122] Paul Shaw. “Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems”. In: 1998, pp. 417–431. DOI: 10.1007/3-540-49481-2{_}30.

[123] Stefan Ropke and David Pisinger. “An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows”. In: *Transportation Science* 40.4 (Nov. 2006), pp. 455–472. ISSN: 0041-1655. DOI: 10.1287/trsc.1050.0135.

[124] Thomas Chabot et al. “Order picking problems under weight, fragility and category constraints”. In: *International Journal of Production Research* 55.21 (Nov. 2017), pp. 6361–6379. ISSN: 0020-7543. DOI: 10.1080/00207543.2016.1251625.

[125] Heinrich Kuhn, Daniel Schubert, and Andreas Holzapfel. “Integrated order batching and vehicle routing operations in grocery retail – A General Adaptive Large Neighborhood Search algorithm”. In: *European Journal of Operational Research* 294.3 (Nov. 2021), pp. 1003–1021. ISSN: 0377-2217. DOI: 10.1016/J.EJOR.2020.03.075.

A

Gantt Chart

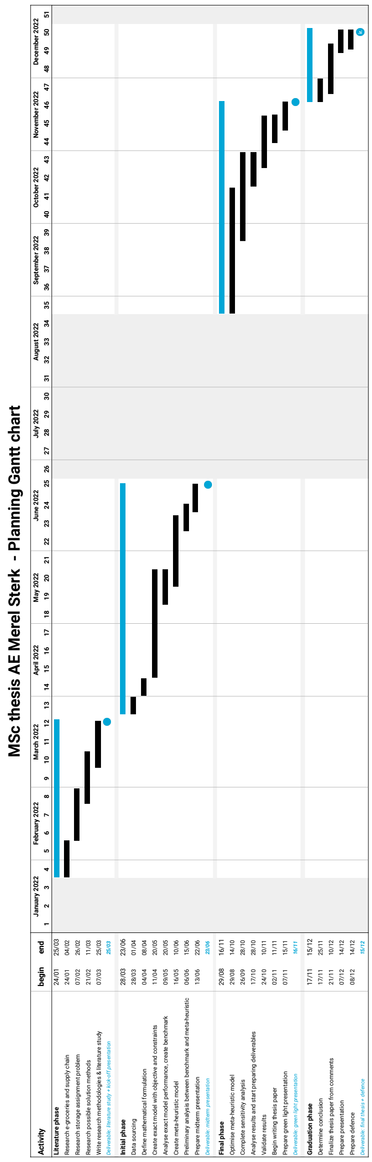


Figure A.1: Gantt chart of the project.

III

Supporting work

1 Model Verification

Before conducting computational experiments for the research, the models should be verified. This entails checking whether the models respond the way they are designed to behave. This is tested with a very small, simplified data set to determine the effects.

There are three scenarios that are verified, being the MSL possibility, the minimisation of travelled distance, and the capacity constraint of a zone.

MSL possibility

The possibility for MSL is embedded into the models and this entails that they should allow for SKUs to be stored in more than one zone if the pick circuit. If the number of MSL that may be present is set to one and it is tried with a very small data set, Figure 1.1 shows that indeed all SKUs are assigned to one location, with one SKU being assigned to more than one location.

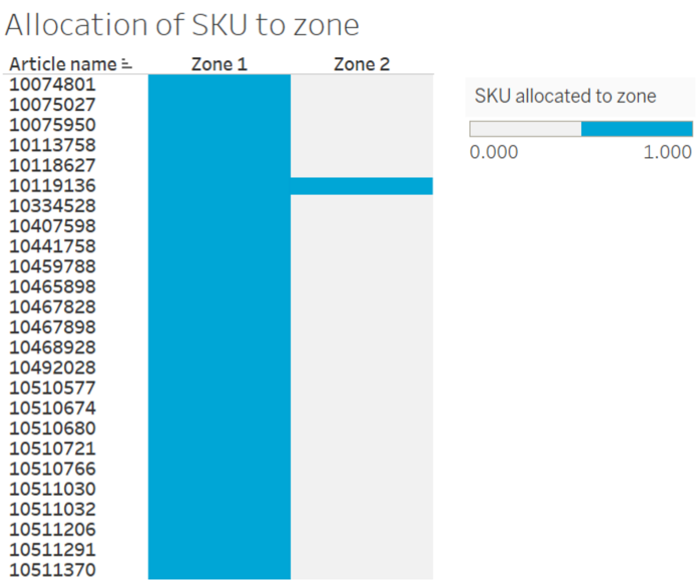


Figure 1.1: MSL of one SKU.

Minimisation of travelled distance

The goal of the models is to minimise the travelled distance of order pickers. This comes down to minimising the number of zones a pick cart has to visit and minimising the total number of pick carts. Therefore, it holds that if there is ample storage capacity in a zone and none in the other, all SKUs should be placed in that zone such that the pick carts only need to visit one zone. Figure 1.2 shows that if all products are placed in one zone, the pick carts indeed only visit that zone. Not the full SKU list is shown for the sake of readability. Also, if there are only a small amount of orders, the orders should be placed on the same pick cart to minimise the travelled distance. That the models adhere to this is evident by Figure 1.2 as well.

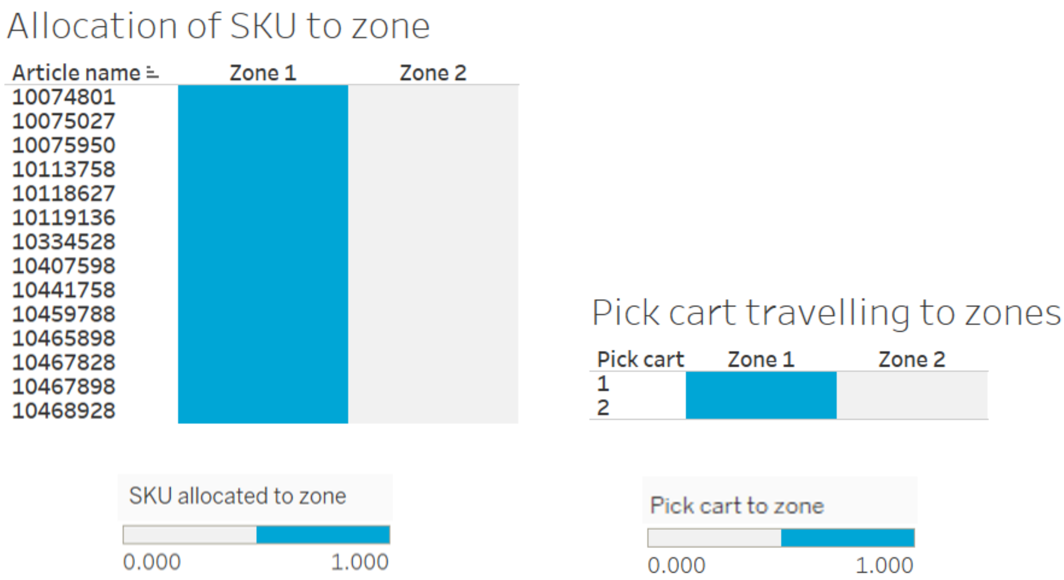


Figure 1.2: Minimisation of travelled distance.

Capacity constraint
The capacity constraint ensures that not more SKUs are placed in a zone than there is space available. If the capacity is very large and there are no restrictions on MSL, Figure 1.3 shows that all SKUs have MSL. However, if the capacity is too small for all SKUs, the models become infeasible which is also desirable behaviour.

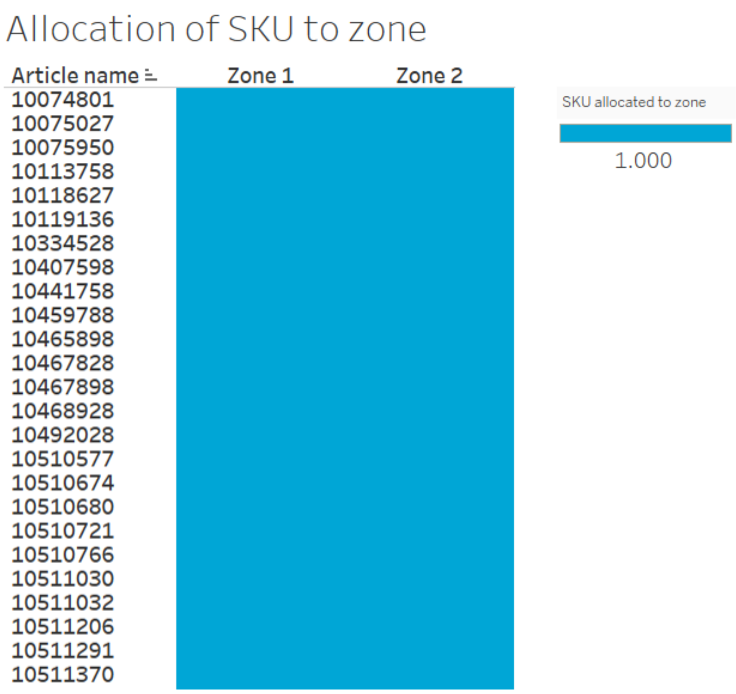


Figure 1.3: Capacity constraint.

2

Case Study

As mentioned in the previous sections of this thesis, a case study was conducted at Picnic Technologies, a Dutch e-grocery retailer. Picnic provided a data set which is explained in section 2.1. Even though this research only focused on the processes related to order picking, there are other processes that are influenced as well. How they are influenced and which matters must be taken into account is discussed in section 2.2.

2.1. Picnic Data Set

The data set from Picnic is obtained from their Data Warehouse (DWH) and queried via Structured Query Language (SQL). Two data sets are used, one for all data of the orders and one for all data of the SKUs. The respective examples of these data sets can be seen in Table 2.1 and Table 2.2. Please note that the data in these examples is not real.

Table 2.1: Data set (simplified and randomised) example of order data, obtained from Picnic DWH.

Order ID	SKU ID	Truck ID
1	15	1
1	23	1
1	4	1
2	8	1
2	2	1
3	15	2
3	8	2

Table 2.2: Data set (simplified and randomised) example of SKU data, obtained from Picnic DWH.

SKU ID	Name	Aisle	Height	Width	Length	Front surface	Demand	Storage type	Storage factor	Storage multiply factor
1	Komkommer	D	5	5	30	25	325	1	975	0
2	Toiletpapier 3laags	A	30	40	10	1200	212	1	975	0
3	Pasta penne 500 g	B	10	5	4	50	105	2	0	0.05
4	Volkoren brood	C	9	13	28	117	89	3	0	0.05
5	Pepsi Cola 6fl	A	23	8	8	184	126	1	975	0

In reality, Picnic offers more than 8000 different SKUs and certain FCs can handle up to 5000 orders per day. Picnic has multiple FCs throughout several countries, yet their generic set-up is the same. If the models provide results for one FC, they can provide results for all Picnic FCs provided that some FC-specific characteristic parameters are changed. Setting up a live connection between the models and the DWH with automatic pre-processing of the data would enable operational leadership to change the SKU allocation on any given time. For example, the (two-)weekly order forecast could be used in the models to give the best SKU allocation.

Table 2.2 shows that there are multiple storage types. These are mentioned in the scientific paper as being (i) pallets, (ii) shelving units, (iii) dollies, (iv) flowracks, and (v) roll containers. The combination of these five types might be unique to Picnic, however the models can handle more and/or different combinations of storage types. These storage types are important when it comes to determining the needed capacity for storing all the SKUs. It is set in such a way that if a SKU is stored on a specific storage type, such as a pallet, it takes up a predefined amount of space. However, if a SKU is stored on a flexible storage type, such as a shelving unit, the amount of space it takes up is determined by its demand and volume. Figure 2.3 through Figure 2.5 shows the different storage types.

Figure 2.1: Pallets.

Figure 2.2: Shelving units.

Figure 2.3: Dollies.

Figure 2.4: Flowracks.

Figure 2.5: Roll containers.

2.2. Influence on Other Operational Processes

When looking at the SAP, OPRP, and OBP, the situation as laid out by the models is very similar to the current operational situation in the Picnic FCs. One big difference and new initiative would be the introduction of MSL. MSL would entail a multitude of changes to the WMS of a Picnic FC and also to some operational processes. Because of this reason, MSL are not yet implemented or tested into an FC, hence there are no empirical results. During the research all needed changes were assessed and ideas for implementation were created. This section discusses the changes on the order picking side and the replenishment side.

2.2.1. Order Picking Changes

During the order picking process there are some changes in the WMS when introducing MSL. This happens at two stages: during the planning of the pick round and during the actual pick round.

During the planning of the pick round it needs to be determined from which location a MSL SKU is taken. The goal is to minimise the travelled distance, hence the optimal location of the SKU should be chosen and that is dependent on the other SKUs in the order. One thing that does come into play is the current amount of stock. If during the planning stage of the pick round there is not enough stock to pick the SKU from a certain location, that location is not considered possible to visit. This is done in order to prevent sending order pickers to a location where there is no stock.

During the pick round the route is already set. When a SKU has MSL and multiple of them are passed, it needs to be determined from which location the SKU is actually picked. This is determined by the *picking category* of the MSL SKU and the other SKUs in the order. There is an order of picking categories, the *pick sequence*, which ensures that heavy SKUs are placed before fragile SKUs. The MSL SKU is picked from the location where it does not interfere with the pick sequence. This is most often the primary location. If the case presents itself that there are multiple locations that the SKU can be picked from and there is no difference between them, the SKU will be taken from the location that has the most stock at that moment. Despite these measures, the scenario can occur that an order picker is sent to a location where there is no stock available. If the other location is still to be passed in the same pick round, the WMS needs to adjust for this fact and add the pick at the later location.

2.2.2. Replenishment Changes

Replenishment is the restoration of a SKU to a former level or condition [3]. Replenishment of SKUs that have one location is straightforward: move the incoming stock to the storage location and anything that is superfluous gets moved to a so-called buffer location. This changes when a SKU has MSL, since there needs to be a divide made between all incoming stock. This divide is dependent on the expected number of picks per location of the SKU and is based on a forecast. An additional challenging part of this is when SKUs are highly perishable and the Best-Before-Date (BBD) must be taken into account. The challenge of the BBD will not be tackled in the first version of WMS adjustments to MSL and is therefore not discussed further.

A heuristic is developed to decide which quantity of the stock goes to which storage location in case a SKU has MSL. The heuristic has two parts, based on whether or not the demand for a location is known. The *semi-urgent demand* is where the number of orders needed to pick the SKU and the composition of the orders is known. *Non-urgent demand* is when there is no information on orders and demand. For both these cases the concept of a *primary location* is introduced. The primary location is the location where if there is a choice, the SKU is picked from. This primary location is based on the picking category of the SKU. For example, bread is a very fragile SKU and it is better to pick this SKU near the end of the pick round. Then the bread is not smashed by other, heavier SKUs. Therefore, if bread is present at MSL, the primary location will be the location that is closest to the end of the pick circuit.

Semi-urgent demand

1. Determine for all non-MSL SKUs from which zone in the pick circuit they are taken.
2. If all non-MSL SKUs are taken from the same zone, determine that the MSL SKU also is taken from that zone.
3. If all non-MSL SKUs are taken from different zones, determine that the MSL SKU is taken from the primary location.
4. Set the number of picks as the demand, determine the demand split and use the same split for the stock.

Non-urgent demand

1. Set the initial demand for the primary location at 15%.
2. Determine, based on historic data, for each zone the number of pick rounds that only travel to that specific zone.
3. Determine, based on the same historic data, the number of pick rounds that travel to more than one zone.
4. Set the number of visits to a specific zone as the demand, with 15% additional demand for the primary location. Determine the demand split and use the same split for the stock.
5. Any leftover stock is distributed evenly over all locations.

This heuristic should ensure that a SKU can always be picked from the best possible location in terms of reducing travelled distance and that the order pickers should not encounter empty storage locations. First, the semi-urgent demand is handled and if enough stock has come in, the non-urgent demand is also handled.

Bibliography

[1] René de Koster, Tho Le-Duc, and Kees Jan Roodbergen. Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, 182(2):481–501, 10 2007. ISSN 03772217. doi: 10.1016/j.ejor.2006.07.009.

[2] Arkadiusz Kawa. Fulfillment service in e-commerce logistics. *Logforum*, 13(4):429–438, 12 2017. ISSN 18952038. doi: 10.17270/J.LOG.2017.4.4. URL http://www.logforum.net/volume13/issue4/abstract-4.html.

[3] Chan-Ju Lee and Suk-Chul Rim. A mathematical safety stock model for ddmrp inventory replenishment. *Mathematical Problems in Engineering*, 2019, 2019.

[4] Seidel Saskia, Nora Mareï, and Corinne Blanquart. Innovations in e-grocery and logistics solutions for cities. *Transportation Research Procedia*, 12:825–835, 2016.