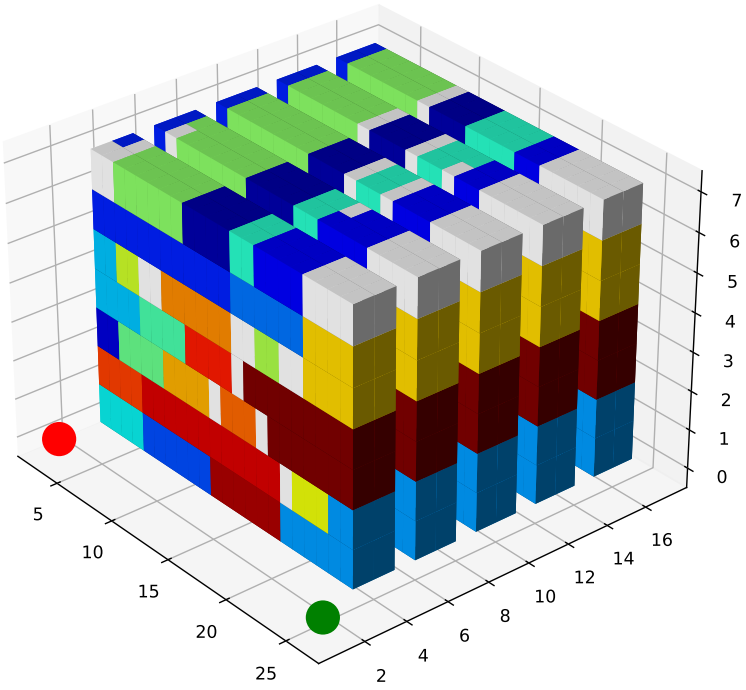

Improving Warehouse Slotting Using Clustering and Genetic Algorithm

Master's thesis in Complex Adaptive Systems, and Supply Chain Management

NICOLE ADAMAH
FILIP LINNÉR

DEPARTMENT OF TECHNOLOGY MANAGEMENT AND ECONOMICS
DIVISION OF SUPPLY AND OPERATIONS MANAGEMENT

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024
www.chalmers.se

Improving Warehouse Slotting Using Clustering and Genetic Algorithm

NICOLE ADAMAH
FILIP LINNÉR

Department of Technology Management and Economics
Division of Supply and Operations Management
Chalmers University of Technology
Gothenburg, Sweden 2024

Improving Warehouse Slotting Using Clustering and Genetic Algorithm

NICOLE ADAMAH
FILIP LINNÉR

Department of Technology Management and Economics
Chalmers University of Technology
SE-412 96 Gothenburg
Sweden
Telephone +46 (0)31 772 1000

Cover:

Visualization of cluster placement when optimizing a fictional warehouse layout for the best possible ergonomics.

Gothenburg, Sweden 2024

Improving Warehouse Slotting Using Clustering and Genetic Algorithm
NICOLE ADAMAH
FILIP LINNÉR
Department of Technology Management and Economics
Chalmers University of Technology

Abstract

The optimization of warehouse operations, particularly order picking, is crucial for reducing operating costs and enhancing ergonomics to prevent work-related injuries. This thesis addresses the challenge of optimizing slotting in manually operated warehouses by integrating clustering and genetic algorithms to improve order picking efficiency and ergonomics. Using K-Medoids clustering, products were grouped into clusters, which were then strategically placed in the warehouse through a genetic algorithm to minimize picking distance and improve ergonomic conditions. The study further refined slotting by optimizing the placement of products within clusters. The results demonstrate that this AI-driven approach outperforms random slotting schemes, significantly reducing picking distance and enhancing ergonomic safety. Moreover, clustering using AI methods produces more well-defined and evenly distributed clusters compared to traditional ABC analysis. The study highlights the importance of the clustering function's logic in achieving optimal warehouse slotting and suggests that a well-designed AI-powered slotting system can lead to substantial operational improvements. A quantitative case study method was employed to test the algorithm, confirming its effectiveness in a real-world setting. This research contributes to the field of warehouse management by demonstrating the effective integration of AI in slotting optimization. The findings provide valuable insights that can be applied across various industries, paving the way for more intelligent and effective supply chain solutions.

Keywords: artificial intelligence, warehouse, supply chain, clustering, slotting, genetic algorithm

Acknowledgements

We would like to thank the team at DB Schenker Consulting for their valuable insights into warehousing operations and their practical advice during the course of this thesis. Especially we would like to thank our supervisor at DB Schenker Consulting, Ida Falkeby for the great help provided during the journey of the project. We also extend our gratitude to our supervisor and examiner at Chalmers University of Technology, Tarun Agrawal, for his guidance, thoughtful feedback, and support in refining our work.

Nicole Adamah & Filip Linnér, Gothenburg, June 2024

Contents

List of Acronyms

In the list below, acronyms are presented that have been commonly used in the report.

AI	Artificial Intelligence
GA	Genetic Algoritm
SKU	Stock Keeping Unit
SLAP	Storage Location Assignment Problem
SOA	Stochastic Optimization Algorithm

List of Figures

List of Tables

1
Introduction

The first chapter of this thesis provides the background of the topic, outlines the purpose of the report, and details the research questions and limitations. This sets the foundation for the study and outlines the direction of the research.

1.1 Background

In recent years, artificial intelligence (AI) has gained significant popularity due to the widespread use of applications such as ChatGPT (Marr, 2023). AI has become a crucial tool not only for private individuals but also for businesses. For instance, over 80% of business executives view AI as a key driver for growth, while more than 70% fear losing their competitive edge if they do not scale up their AI usage (Reilly et al., 2019).

Simultaneously, recent investments in supply chains have increasingly focused on cost reduction and speed enhancement (Smith, 2022). A major strategy for cutting costs involves optimizing warehousing operations, thereby reducing the need for resources.

The convergence of AI and warehousing cost reduction presents a unique challenge for modern businesses. Given the complexity of supply chains and the vast amounts of data they generate, this thesis aims to utilize AI to improve warehouse efficiency.

1.1.1 Warehousing

In 2020, there were over 150,000 warehouses globally, with the number expected to grow by almost 20% by 2025 (Placek, 2022). It is estimated that companies spend over $350 billion on warehousing annually, a figure that is also projected to increase (Bustamante et al., 2020). One of the leading risks for large companies is business interruptions, including supply chain disruptions (Rudden, 2024). Warehouses play a crucial role in mitigating these interruptions by ensuring efficient operations. Efficient warehouse operations help manage direct costs associated with the supply chain and reduce indirect costs stemming from potential interruptions, such as delays in production and delivery, loss of sales, and customer dissatisfaction. Factors contributing to these interruptions include equipment failures, inventory mismanagement, and inadequate storage solutions, all of which underscore the necessity for

optimized warehouse management practices (Gu et al., 2007).

Operating a warehouse involves several tasks, such as receiving, storing, and picking stock-keeping units (SKUs) (Gu et al., 2007). Evaluating the storage function requires considering two main areas: access efficiency (how much resources are needed for picking SKUs into an order) and storage efficiency (how many SKUs can be stored in the warehouse). This evaluation leads to the storage location assignment problem (SLAP). SLAP is a challenge that involves allocating SKUs to storage positions in a way that maximizes space utilization and minimizes material handling costs. Solving SLAP is crucial for enhancing warehouse efficiency and reducing operational costs.

Looking at the costs of warehousing reveals that order-picking can account for over 50% of operating expenses (de Koster et al., 2007; Pang & Chan, 2017), with travel within the warehouse consuming approximately 50% of the picker's total time (Tompkins et al., 2003). This underscores the significant potential for cost savings through enhanced picking efficiency, which can partly be achieved by optimally assigning SKUs to storage locations to minimize travel distance. Effective SKU allocation is fundamental for route optimization, as routing decisions depend on the location of stored items (R.-Q. Zhang et al., 2019).

Another important aspect, which is somewhat foreseen within warehouse operations, is the ergonomics for the warehouse workers. This is still the reality even when over 60% of work-related health issues in the European transportation and storage sector are connected to muscular problems (Calzavara et al., 2017). These health problems not only affect individuals but also impose costs on companies. One common muscular problem is low back disorder, which historically has accounted for over 15% of the compensations claimed by workers in general. Although the number of cases are around 15%, the costs for these reach up towards 40% of total costs for work-related health issues (Marras et al., 1999).

1.1.2 Artificial Intelligence

The term artificial intelligence was introduced in the mid-1950s (Anyoha, 2017). At that time, computers were not capable of running advanced AI programs, which limited the pace of AI development. However, thanks to Moore's law, computers have exponentially increased in speed and memory capacity, making it possible to execute advanced AI programs today. In recent years, several AI-based tools have been launched, such as ChatGPT (Marr, 2023), Microsoft Copilot (Spataro, 2023), AI services within Amazon Web Services (Amazon Web Services, n.d.), and AI-supported warehousing systems (Swisslog, n.d.).

Even though AI has been used more widespread in the last years, the public is having more concerns than excitement for the technology (Rainie et al., 2022), which could impose a hinder for even greater utilization of the technology. One example is

that more people think that driverless passenger vehicles are a bad idea for society rather than a good one.

Taking the perspective of AI within supply chain management reveals three main branches, namely *sensing and interacting*, *learning*, and *decision making* (Pournader et al., 2021). Sensing and interacting refers to AI systems that interact with human input. The learning-branch is about using data for training algorithms to solve new problems. The final branch, decision making focuses on solving optimization issues.

AI in supply chain management can be categorized into three main branches: *sensing and interacting*, *learning*, and *decision making* (Pournader et al., 2021). Sensing and interacting involves AI systems interacting with human input. The learning branch uses data to train algorithms to solve new problems, while the decision-making branch focuses on solving optimization issues.

From a broader perspective, the purpose includes improving worker health and reducing product costs in warehouses, which can have positive societal impacts. Ethically, this thesis advocates for warehouse optimization that considers both resource efficiency and the working environment, contributing to a more human-focused approach. Environmentally, better slotting can reduce picker travel distances and, consequently, lower the energy required to operate the warehouse, thus conserving resources.

1.2 Purpose

The purpose of this thesis is to investigate how AI solutions can optimize slotting to enhance warehouse operations, particularly by improving the working environment and reducing order picking distances. By examining these areas, the thesis aims to provide insights into using AI for slotting and offer guidance on designing tools for warehouse optimization. Additionally, it seeks to identify potential challenges in implementing AI-supported slotting optimization.

From a broader perspective, this thesis aims to provide a fresh perspective on slotting by emphasizing ethical considerations and focusing on worker well-being. It seeks to highlight the importance of optimizing warehouse operations not only for resource efficiency but also for improving the working environment. This human-centered approach is often overlooked in current warehouse optimization literature. Additionally, by improving slotting to reduce picking distances, the thesis demonstrates how such optimizations can lead to lower energy consumption, thus contributing to more sustainable warehouse operations.

1.3 Research Questions

As this thesis focuses on the intersection between artificial intelligence and warehouse management, the following research questions were chosen:

RQ 1 How can AI-driven solutions optimize the storage location assignment problem (SLAP) to reduce travel distance, while optimizing the ergonomics for workers?

This question aims to investigate the potential of AI to improve operational decisions within warehousing by not only improving the operational aspects in the form of picking route distance but also by considering the ergonomics for the workforce. The motivation behind this question lies in addressing the dual challenge of increasing efficiency and reducing work-related health issues within the warehousing environment.

RQ 2 What are the challenges and considerations in implementing AI-supported slotting optimization solutions in warehouses?

This question seeks to explore the practical aspects of implementing AI technologies into existing warehousing operations, including the identification of potential obstacles. The motivation for this question is to provide insights and guidelines for businesses aiming to leverage AI for warehousing optimization, ensuring a smooth transition and maximization of benefits while minimizing disruption and resistance.

1.4 Limitations

Working on the thesis has had limitations. This section will present the limitations that has been present and describing them briefly.

The first limitation of this study is the exclusive focus on travel distance for the picking operation. While various metrics, such as time spent on picking, could have been considered, the analysis was limited to travel distance to ensure it was independent of the type of vehicle used for transporting the picker. This approach aimed to maintain consistency and comparability across different scenarios, avoiding variability introduced by different types of picking equipment.

The second limitation is about the layout of the warehouse that is being analyzed. To ensure the solution is applicable to various environments, some simplifications had to be made with respect to the layout that is being analyzed. In general, most warehouses can be seen as rectangles, meaning that using simple geometrical shapes will often represent most of warehouse layouts.

The third limitation involves routing heuristics. Since the focus is on the location of SKUs rather than the routes pickers take, simple routing heuristics were used

instead of more complex, optimal routing.

The fifth limitation is that only one firm will be used for the case study. This means that a potential solution only will be tested within a single environment. Potentially, this could result in a solution that usable only in this environment, but to hedge for this risk, two years of data is available from the case company, meaning that the solution can be analyzed twice to verify its usefulness.

A sixth, and final, limitation is about creating a user-friendly application. Since the purpose is to investigate how an AI-solution can enhance the warehousing operations, the focus is not on creating a final program to be implemented during the phase of warehouse analysis, but rather to create a more basic tool or program that can be used if having the correct information about using it. In other words, a finished application is not created, but rather a simpler script.

2
Theory

To provide a theoretical basis of the thesis, this chapter presents relevant theory for the purpose of the study. It is structured into three main parts. The first part describes how the literature search was made, the following part describes how warehousing works, and the final part is about the current landscape of research within the joint field of warehousing and artificial intelligence with presentations of common AI methods used in previous research.

2.1 Literature Review

The initial step of the research consisted of a reviewing the literature connected to the research area. This step ensured an understanding of existing theory and that all relevant publications connected to the subject where covered (Pruzan, 2016).

The research scope was defined through the selection of keywords that accurately reflected the research topic, the chosen keywords were ”warehouse AND artificial OR intelligence”. These keywords served as the foundation for a search executed in the *Scopus* database. The initial query, detailed in Figure 2.1, resulted in a collection of around 6,000 articles. To refine this data collection and enhance relevance, the search parameters were narrowed to include only articles published within the last 5 years, written in English, and directly related to the specified research area. The final search resulted in just over 400 articles (in February 2024), which were then evaluated based on relevance, ensuring that only those articles that significantly contributed to the research topic were retained. This systematic procedure was made to effectively cover the majority of published articles on the topic, as highlighted by Bell et al. (2019).

```
TITLE-ABS-KEY ( warehouse AND artificial OR intelligence )
   AND PUBYEAR > 2017 AND PUBYEAR < 2025 AND ( LIMIT-TO (
   SUBJAREA , "COMP" ) OR LIMIT-TO ( SUBJAREA , "ENGI" ) OR
    LIMIT-TO ( SUBJAREA , "MATH" ) OR LIMIT-TO ( SUBJAREA ,
    "DECI" ) OR LIMIT-TO ( SUBJAREA , "BUSI" ) OR LIMIT-TO
   ( SUBJAREA , "ECON" ) ) AND ( LIMIT-TO ( DOCTYPE , "ar"
   ) ) AND ( LIMIT-TO ( LANGUAGE , "English" ) )
```

Figure 2.1: Code describing the search words used for literature search in Scopus.

After establishing the base literature for the work, a more intuitive approach was used to identify other relevant sources. More keywords were used together in Scopus, resulting in more narrow results. Example of keywords were *supply chain*, *cluster*, *slotting*, *genetic*, and *storage location assignment problem*. Outside of the main literature search, there was no limitation of the publishing year of the sources since some areas, such as worker health, had commonly cited articles that were far out of the above used publishing time frame.

Additionally, a backward search method was used to discover further literature. This involved reviewing the references of key articles to identify other relevant sources (Ermel et al., 2021).

2.2 Warehousing

Warehousing is defined as *"the activity of storing something in a warehouse"* (Cambridge Dictionary, n.d.). Simplifying the activities involved, a warehouse requires a layout of racks and storage systems, slotting SKUs to these racks, and picking SKUs to fulfill orders. Poor warehouse operations negatively impact customer service and business performance in terms of service quality and operating costs (Rushton et al., 2021). It is of importance to understand that the decisions made for the layout have an impact on the way the slotting can be done, and the slotting impacts how the picking is done in the most effective way (de Koster et al., 2007). Since this report focuses on improving picking efficiency through more effective slotting, the primary emphasis will be on slotting and picking, with a brief overview of warehouse layout.

2.2.1 Layout

A key aspect of warehouse design is its layout, which includes the size and shape of the space, location of departments, and the arrangement of storage systems such as shelves and racks (de Koster et al., 2007). The primary goals of designing a warehouse layout are to ensure rational design and maximize space utilization (Jonsson, 2008). Consequently, this requires balancing space utilization with efficient travel paths within the warehouse. High space utilization typically reduces travel distances for picking and vice versa.

Regarding storage systems, two main ways of storing goods is by using either shelves for smaller goods that can be picked by hand, or to use racks, which store pallets of goods (Jonsson, 2008). Both of these solutions can handle multiple layers of products, allowing items to be picked from different heights and not just the top layer.

2.2.2 Slotting

When SKUs arrive at the warehouse they need to be stored somewhere, which can be called slotting. There are five main policies are traditionally used to assign SKUs

to the positions in the warehouse, these are *random storage*, *closest open location storage*, *dedicated storage*, *full-turnover storage*, and *class-based storage* (de Koster et al., 2007).

Random storage is simple. The goods are just spread out randomly around the warehouse (de Koster et al., 2007). The benefit of this policy is that the space utilization has potential of being high. The downside is that the travelling distances will increase.

Closest open location storage involves placing products in the nearest available spot in the warehouse (de Koster et al., 2007). This policy performs similarly to random storage, especially for full pallet loads.

Dedicated storage is more precise than the two previously mentioned policies. This is since each product has its own location, independent if it's out of stock or not, which is one of its downsides (de Koster et al., 2007). An upside with this policy is that picking efficiency can potentially be enhanced due to pickers getting to know the exact locations of each SKU.

Full-turnover storage prioritizes placing frequently sold products in easily accessible slots (de Koster et al., 2007). This policy requires more data than the previously described policies, which is also one of its drawbacks. For example, when locations are determined by demand, fluctuations in demand can rapidly make the policy inefficient, requiring a potentially costly reorganization of the SKUs to achieve optimal efficiency again.

Class-based storage combines elements of the previously mentioned policies, often creating three classes based on turnover. Typically, 15% of the SKUs account for 85% of the turnover. These high-turnover products are grouped into the same class and assigned a dedicated storage area. Within this area, products are randomly allocated (de Koster et al., 2007). This process is repeated for lower-turnover products. While travel distances are slightly longer compared to full-turnover storage, class-based storage is easier to implement.

2.2.3 Picking

Once SKUs have been slotted and a picking order is received by the warehouse, it's time to initiate the picking process. A picking order is built by several order lines, with each line representing a single SKU and the quantity to be picked (de Koster et al., 2007). Therefore, a picking order consists of several order lines (if the order has several SKUs) describing to the picker what products to picks and how many of them. In traditional warehouses, where pickers retrieve items from racks, the system is referred to a *picker-to-parts* system (de Koster et al., 2007).

Order lines can be split between pickers in two primary ways (de Koster et al., 2007).

The first method, known as *discrete picking*, involves a single picker completing an entire order. This means that the picker gathers several order lines until the entire order is fulfilled. The second way, *batch picking*, assigns one picker a few items from different orders. While this results in higher quantities of each SKU being picked, the picker visits fewer SKUs per route.

Picking Routes Regardless of the order line split method being used, determining the movement of pickers within the warehouse is crucial. Efficiently collecting SKUs is important because staffing costs in a conventional warehouse account for approximately 50% of total expenses (Rushton et al., 2021). The routing of warehouse pickers resembles the traveling salesman problem, where the picker starts at one location, visits each picking spot once, and then proceeds to the goods release point (de Koster et al., 2007). While algorithms exist to solve this problem, in practice, simpler heuristics are often employed.

One common heuristic is the *S-shape* method (de Koster et al., 2007). This approach involves visiting every aisle in the warehouse that contains an SKU to be picked, following a route that resembles the letter "S." Specifically, the picker moves up one aisle and down the next. Figure 2.2 provides a visualization of this heuristic.

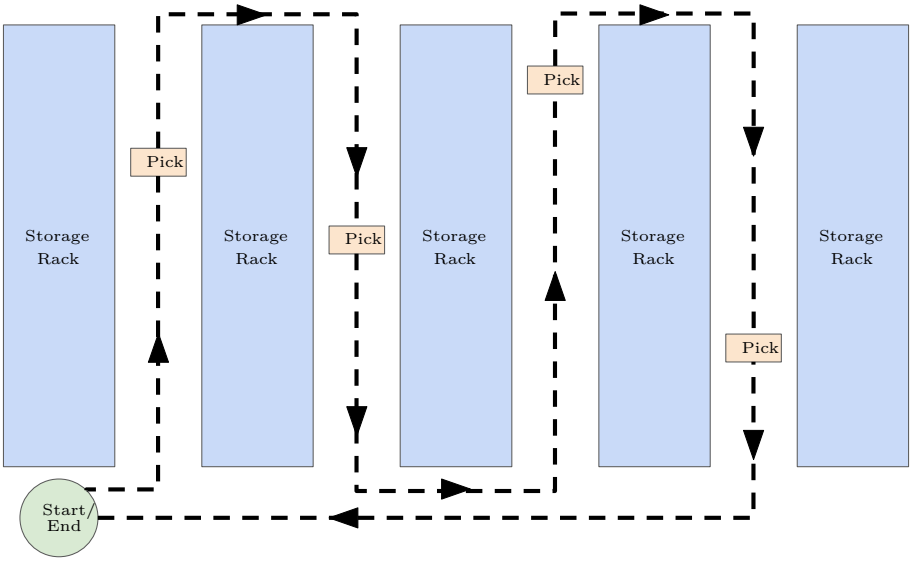

Figure 2.2: Visualization of the *S-shape* picking route heuristic, own adoption from de Koster et al. (2007).

Another heuristic is the *return method* (de Koster et al., 2007). This approach starts at one end of the aisles and moving up each aisle until all required SKUs have been picked. By placing high-demand SKUs close to the starting point, the picking distance is minimized. Figure 2.3 illustrates the return method heuristic.

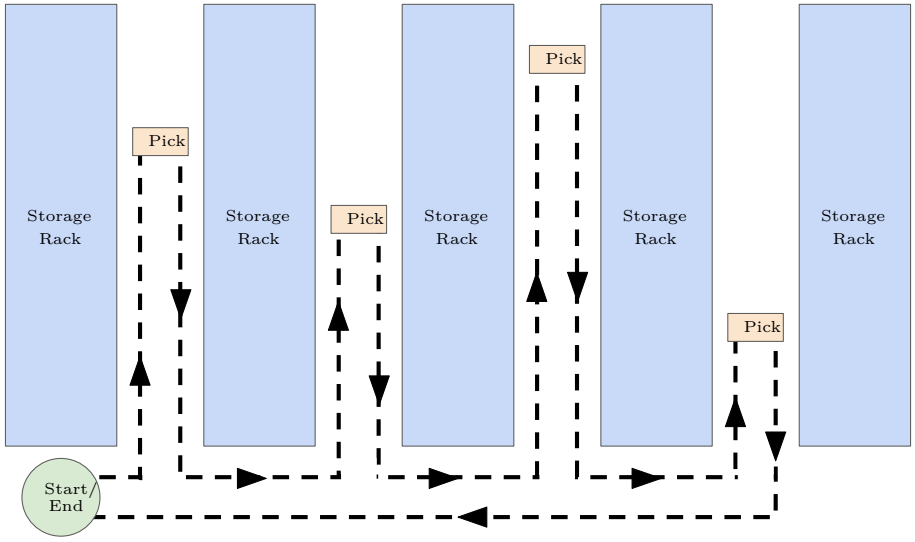

Figure 2.3: Visualization of the *return method* picking route heuristic, own adoption from de Koster et al. (2007).

2.2.4 Health Considerations for Warehouse Workers

Warehouses, unless fully automated, require human workers for operation. In a parts-to-picker system, workers must move around to conduct picking. Conventional warehouse operations show that around 20-25% of the total operational cost comes from picking and packing orders (Rushton et al., 2021). However, cost is not the only concern for workers. Heavy products can strain workers, leading to low back disorders (Marras et al., 1999). Manual material handling, common in warehouse picking, is a major cause of these disorders. This issue not only bad for workers, but also for the companies of where these problems occur. For example, within the European transport and storage sector, musculoskeletal disorders accounted for 62% of the health problems related to work (Calzavara et al., 2017). Marras et al. (1999) presents that between 33 and 41% of all compensation claims made by workers are related to low back disorders.

Experiments by Marras et al. (1999), where six different locations of lifts were performed highlighted that the lifts that had the most impact on spine compression (causing low back disorders) were all at low vertical height and either close to the feet of the lifter or far away from the lifter. The best locations where at a high or medium vertical height close to the lifter. Additionally, the study found that the measurements made on the spine increased by around 15% for every 4.5 kilograms of added weight to the lift.

Another study (Calzavara et al., 2017) introduced the concept of energy expenditure in relation to ergonomics and fatigue. Which means that the energy needed for lifts must be considered when optimizing warehouse ergonomics. Taking a physics perspective of this reveals that it could be seen as something similar to the equation for work ($W = f \cdot d$), with the total work (W) is determined by the forced applied (f) and the distance (d) that the force is being applied on (Britannica, 2023). This high-

lights that many light lifts can cause long-term effects, not just infrequent heavy lifts.

Research has been done for determining weight limits for lifts (Waters et al., 1993). This study introduces an updated equation for this purpose. While not delving into the specifics of the equation, the paper outlines several general factors that negatively impact lifts. Firstly, horizontal distance from the operator worsens lifts. Secondly, lifts that are far from the picker vertically, are also considered worse than lifts close to the operator. Thirdly, long-distance lifts are worse than shorter ones. Fourthly, asymmetrical lifts are more harmful than symmetrical ones. Fifthly, adding handles to boxes reduces strain. Lastly, lift frequency affects the maximum allowed weight, with higher frequencies reducing this weight.

2.2.5 ABC-Analysis

One way to analyze the products stored in a warehouse is by using an ABC-analysis (Jonsson, 2008). This analysis helps identify key products that significantly impact operations, typically based on total sales or picking frequency. Often, only a small part of items contribute to the majority of, for example, sales volume. Conducting an ABC-analysis involves straightforward steps: first, selecting or calculating the parameter for analysis, then sorting products from highest to lowest based on this parameter, and finally calculating the cumulative sum of the parameter over the items and its share of the total sum. The next step involves assigning A, B, and C-classes to products. Generally, A-class represents products constituting 80% of the cumulative sum, B-class represents items from 80% to 95% of the sum, and C-class include the remaining products. The example provided in Table 2.1 illustrates the application of ABC-analysis to the picking frequency of a specific item set.

Table 2.1: ABC-analysis of a set of items as an example to illustrate the method where the picking frequency is used as the variable for the analysis.

Part Number	**Frequency**	**Cumulative Frequency**	**Share of Cumulative Frequency**	**Class**
34	2600	2600	63.5%	A
78	700	3300	80.6%	B
23	450	3750	91.6%	B
89	150	3900	95.2%	C
01	100	4000	97.7%	C
10	50	4050	98.9%	C
56	45	4095	100%	C

2.3 Artificial Intelligence in Warehouse Slotting

From the selected literature (see Table A.1 in Appendix A), it is evident that various approaches have been explored. Some studies have used the titles of articles as an initial method for clustering SKUs when no other correlations are available

(Xin et al., 2019), while others have employed Artificial Neural Networks (ANN) to classify articles according to ABC-classes (Veres, 2023). Some research has focused on creating hybrid solutions, utilizing multiple types of algorithms to achieve more efficient outcomes than using a single algorithm, as seen in (Jiang et al., 2021).

The common method in these studies (see Appendix A.1) typically involves starting with a literature review, followed by the development of an algorithm or approach to solve a specific problem. This proposed solution is then tested on data, either from real-world scenarios or synthetic sources. By applying these solutions to data, researchers can compare the optimization outcomes to traditional slotting methods.

Previous literature has highlighted several opportunities for using AI in slotting. These opportunities include improvements within the algorithms themselves and broader insights into the field of slotting. For instance, understanding correlations related to demand is crucial for optimizing warehouse slotting (R.-Q. Zhang et al., 2019). Another notable opportunity is incorporating a combination of economic and ergonomic factors in slotting optimization (Lesch et al., 2023). This demonstrates the potential to use diverse criteria in enhancing slotting processes. Additionally, several studies (Li et al., 2016; Schuhmacher & Hummel, 2023; Xin et al., 2019; Yang & Nguyen, 2016) have explored clustering techniques to group products with similar characteristics. Defining cluster variables before conducting clustering can help manage the stochastic nature of methods like K-Means clustering.

However, the literature also identifies several challenges. For example, Schuhmacher and Hummel (2023) noted that ergonomic considerations were often neglected in optimization processes. Other studies (Jiang et al., 2021; Silva et al., 2022; S. Zhang, Fu, Chen, & Mei, 2020) have not accounted for many parameters in their optimization models, revealing a gap in the research. Additionally, some sources (Li et al., 2016; Waubert de Puiseau et al., 2022) mentioned that AI-enabled slotting might lack dynamic capabilities, making it less effective when new products are introduced or demand patterns change. Furthermore, the focus on hybrid algorithms is relatively rare in the literature, indicating a need for more research in this area.

In summary, while AI in warehouse slotting has been studied, there are significant gaps in the research. One gap is the use of multiple parameters, such as ergonomics, in slotting optimization. Another gap is the need for hybrid algorithms to create more efficient and effective solutions. A third gap concerns flexibility, specifically how optimized solutions perform when new products and demand dynamics are introduced. However, it is still clear that creating product clusters enhances picking performance in slotting.

2.3.1 Clustering

Unsupervised learning is a type of machine learning that deals with algorithms designed to identify patterns in data without relying on predefined labels. Unlike supervised learning, where models are trained on data paired with the correct answers, unsupervised learning uncover the underlying patterns naturally occurring among data points (Jo, 2021). In unsupervised learning, the core operation is clustering, where data items are grouped based on their similarities without any predefined labels guiding the process. Clustering strives to maximize the resemblance between data items and their respective cluster prototypes while supervised learning such as classification aims to minimize errors between predicted outputs and actual labels (Jo, 2021).

The process of unsupervised learning can be described as an iterative cycle. Firstly, the number of clusters is decided, then the cluster prototypes are randomly initialized acting as an early estimation for the cluster centers. Following this, data points are grouped with the prototype they most resemble, using defined criteria for similarity. These prototypes are then progressively adjusted to more accurate mirror the characteristics of the clustered items, continuing this cycle until convergence (Jo, 2021).

There are several methods when it comes to clustering such as K-Means clustering, Hierarchical clustering, Agglomerative clustering and K-Medoids (Hastie et al., 2009). An overview of the methods can be seen in Table 2.2.

Table 2.2: Comparison of common data clustering methods (Hastie et al., 2009).

Method	**Algorithm Type**	**Data Type**	**Distance Measures**	**Complexity Note**
K-Means	Partitioning	Quantitative	Euclidean distance	Lower, efficient for large datasets
Hierarchical	Hierarchical	Any	Various, including Euclidean	Higher, especially for large datasets
Agglomerative	Hierarchical	Any	Various, including Euclidean	Similar to hierarchical clustering
K-Medoids	Partitioning	Any	Any appropriate distance	Higher than K-Means

K-Means Clustering, one of the most well-known clustering methods, is an iterative algorithm suited for quantitative data. It partitions data into K clusters by minimizing the within-cluster variances. Each observation is assigned to the cluster with the nearest mean, which is recalculated after each assignment. The process repeats until cluster assignments no longer change. K-Means uses the squared Euclidean distance as the dissimilarity measure (Hastie et al., 2009).

K-Medoids clustering, like K-Means, aims to partition data into clusters but with a key difference in its methodology making it more robust to outliers. Instead of using the mean of items in a cluster as the central point (centroid), K-Medoids selects actual data points (medoids) as central elements. It alternates between assigning observations to the nearest medoid and updating medoids to minimize total dissimilarity within clusters. K-Medoids clustering is particularly useful when the dataset contains outliers or when a non-Euclidean distance measure is more appropriate for capturing the dissimilarities between data points. Unlike K-Means, which can only use quantitative variables due to its reliance on squared Euclidean distances, K-Medoids is more versatile and can handle various types of data and dissimilarity measures (Hastie et al., 2009).

Hierarchical clustering is a method of cluster analysis that aims to build a hierarchy of clusters. It works by either merging individual observations into larger clusters (agglomerative) or splitting a single cluster into finer sub-clusters (divisive), based on a specified dissimilarity measure. The process is visualized using a dendrogram, which shows the hierarchical relationship between clusters (Hastie et al., 2009). This method differs from algorithms like K-Means, as it doesn't require specifying the number of clusters beforehand and can reveal natural groupings within the data. The resulting dendrogram, a tree-like diagram showing the arrangement of the clusters, allows users to select the number of clusters by cutting the dendrogram at a desired level. However, it's computationally intensive for large datasets and the resulting hierarchical structure might not always represent the natural organization of the data (Hastie et al., 2009).

Agglomerative clustering is a type of hierarchical clustering used to group objects in a bottom-up manner. Starting with each object in its own cluster, the algorithm iteratively combines the closest pairs of clusters until all objects are in a single cluster, or until a specified stopping criterion is reached (Hastie et al., 2009).

Determining the optimal number of clusters is crucial in cluster analysis, and various evaluation metrics can be employed for this purpose. Three commonly used metrics include the Silhouette score, the Davies-Bouldin index, and the Calinski-Harabasz index (Rachwał et al., 2023).

The Silhouette score measures the compactness and separation of clusters. It assigns a score to each data point based on how close it is to its own cluster's points compared to points in other clusters. A higher Silhouette score indicates better-defined clusters and and the points within them are close together (Rachwał et al., 2023).

The Davies-Bouldin index measures how spread out the clusters are and how well-separated they are from each other. This is done by quantifying the average similarity between each cluster and its most similar cluster, while also considering the average dissimilarity of points within each cluster. A lower score suggests better clustering, with tight and well-separated clusters (Rachwał et al., 2023).

The Calinski-Harabasz index compares the spread of points between clusters to the spread of points within clusters. Higher scores mean that the clusters are well-separated and compact, with clear boundaries between them (Rachwał et al., 2023).

In Figure 2.4, two scenarios are visualized: one on the left with well defined clusters, and one on the right with poorly-defined clusters. The scenario with well-separated clusters would yield a higher Silhouette score, a lower Davies-Bouldin index, and a higher Calinski-Harabasz index.

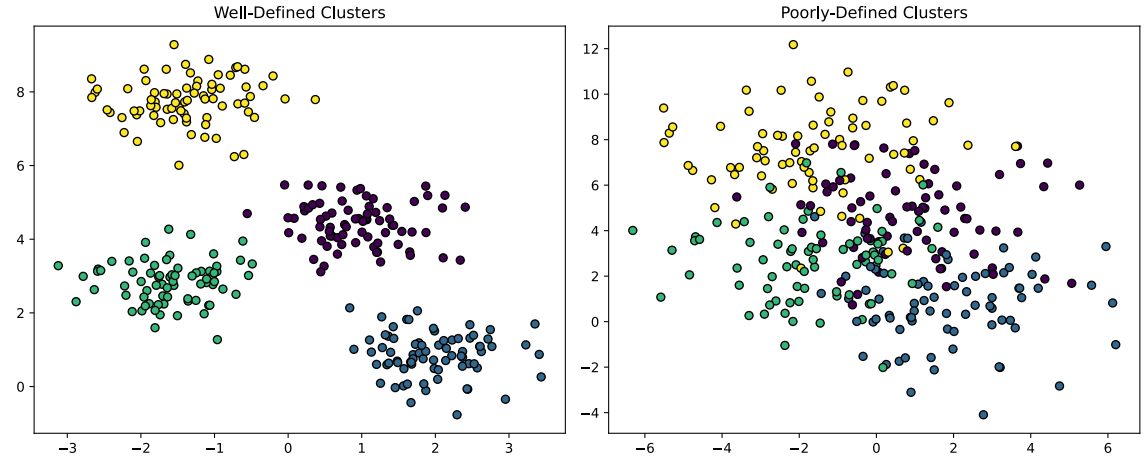

Figure 2.4: Visualization of two clustering scenarios, one with well defined clusters (left plot) and one with poorly-defined clusters (right plot).

2.3.2 Association Rules

As a mean to create more variables for clustering it can be of interest to understand how different variables associate between each other. Finding association rules is seen as unsupervised learning. Furthermore, association rules has become a popular tool in commercial business when mining data (Hastie et al., 2009).
Understanding how different variables associate with each other can create more variables for clustering. Finding association rules is a form of unsupervised learning and is popular in commercial data mining (Hastie et al., 2009).

The Apriori Algorithm The Apriori algorithm is one method for finding association rules (Hastie et al., 2009). It uses items with binary attributes and a database of transactions. Each transaction indicates whether an item has been purchased. The algorithm identifies frequent item sets that meet a certain threshold of occurrences, known as minsupport. It then generates candidate item sets and prunes those that do not meet the frequency criteria. The algorithm's output consists of conjunctive rules (Hastie et al., 2009).
One algorithm that can generate association rules is the Apriori algorithm (Hastie et al., 2009). This algorithm operates on items with binary attributes, denoted as $I_{Base} = I_1, I_2, ..., I_m$. It utilizes a database containing transactions (T), where each transaction (t) indicates whether an item (k) has been bought ($t[k] = 1$) or

not ($t[k] = 0$). The algorithm begins by identifying a subset of items from I_{Base}, referred to as X, and then processes the database to count the frequency of item purchases in each transaction.

The algorithm proceeds by identifying frequent itemsets that surpass a certain occurrence threshold, known as *minsupport*. It then generates candidate itemsets, which are those that appear frequently in the database (T). To enhance performance, the algorithm filter itemsets that do not meet the removes occurrence threshold early in the process. The final output of the algorithm is a set of conjunctive rules (Hastie et al., 2009) that can be interpreted to reveal associations between items.

A key advantage of the Apriori algorithm is its ability to handle large datasets. However, it has limitations in scenarios where the confidence of a rule is high but the support is low. For example, consider the real-world example of vodka and caviar purchases (Hastie et al., 2009). While vodka is frequently bought, caviar is not, yet caviar is often purchased with vodka. In such cases, the algorithm may struggle because the low occurrence of caviar purchases affects its performance.

2.3.3 Genetic Algorithm

A genetic algorithm is an optimization technique inspired by the process of natural selection and evolution. It is commonly used to solve complex optimization and search problems by mimicking the process of natural selection, where the fittest individuals are more likely to survive and reproduce. In a genetic algorithm, a population of candidate solutions (represented as chromosomes or individuals) evolves over successive generations. Each individual in the population represents a potential solution to the optimization problem. The algorithm uses operators such as selection, crossover, and mutation to generate new candidate solutions from the existing ones (Russell & Norvig, 2009). This is illustrated in Figure 2.5.

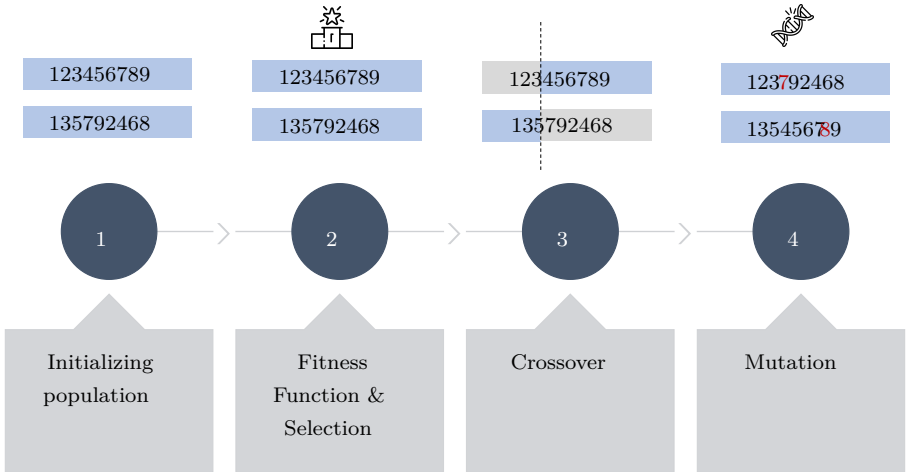

Figure 2.5: Overview of the general steps in a Genetic Algorithm, inspired by an illustration in Russell and Norvig (2009).

One key aspect of genetic algorithms is their stochastic nature. This means that the process of generating new candidate solutions involves randomness or probabilistic

choices. For example, during selection, individuals are probabilistically chosen based on their fitness values, with fitter individuals having a higher probability of being selected. Similarly, crossover and mutation introduce randomness by randomly combining or altering the genetic information of individuals. The stochastic nature of genetic algorithms allows them to explore a wide search space efficiently and find near-optimal solutions to complex optimization problems. However, this also means that the performance of genetic algorithms may vary across different runs due to the randomness involved in the algorithm's operation (Russell & Norvig, 2009).

Initialization The genetic algorithm begins with a randomly generated population of individuals, which are possible solutions to the problem at hand. These individuals, also known as the chromosomes, are typically represented as strings of binary digits, although other encodings are also possible. The initial population size can vary depending on the problem domain and constraints (Russell & Norvig, 2009).

Fitness Function & Selection The fitness function evaluates every individual in a generation based on a set of objectives which from case to case. Based on these evaluations, a selection process is undertaken where individuals are chosen based on their fitness score. The selection strategy might be stochastic but with a higher probability assigned to individuals with greater fitness, encouraging better solution to have a higher change of passing their genes to the next generation (Russell & Norvig, 2009). Techniques as tournament selection, roulette wheel selection, and rank selection are common methods used for selection (Wahde, 2008).

Mutation Mutation introduces genetic diversity into the population by making small random alterations to individual chromosomes. This could be as simple as changing a binary digit from 0 to 1 or vice versa. Mutation ensures that the genetic pool is diverse and helps to prevent premature convergence by introducing new traits for selection (Russell & Norvig, 2009).

Crossover Crossover is a recombination process where two solution is combined, and a new solution is generated. There are many ways to perform crossover, such as single point, which involves picking a random crossover point and exchanging the sequences before and after that point between the two individuals to create new offspring(Russell & Norvig, 2009). Two-point crossover, multi-point crossover and uniform crossover are other variants. The fundamental idea is to combine the genetic information of two parents to produce new chromosomes which may have better fitness(Wahde, 2008).

2.3.4 Hybrid Metaheuristics

The techniques previously described are often utilized for optimization problems. In practical scenarios, achieving a completely optimal solution is frequently unattainable due to computational constraints (Blum & Roli, 2003). Thus, this is where methods that approximate the optimal solution are useful. Metaheuristics involve

combining heuristics with higher-level optimization strategies to efficiently find near-optimal solutions to complex problems by exploring search spaces. These strategies must balance diversification—searching a wide array of potential solutions—with intensification—focusing on promising areas identified from past searches (Blum & Roli, 2003). Examples of metaheuristics include genetic algorithms and ant colony optimization.

The term metaheuristic refers to combining heuristics with higher level optimization to efficiently finding a near optimal solution to a complex problem, which can be described as exploring search spaces. To find these solutions, a metaheuristic must be capable of searching through a diverse range of possibilities while also learning from previous iterations to focus on more promising areas. This process involves balancing diversification, which is the exploration of a wide variety of solutions, and intensification, which is the concentration on the most promising solutions identified so far (Blum & Roli, 2003). Examples of metaheuristics include genetic algorithms and ant colony optimization.

A way to make metaheuristics work even better is by combining several methods, these algorithms are called hybrid metaheuristics. This approach has been shown to perform the best compared to non-hybrid algorithms in practical applications (Talbi, 2002). Hybrid algorithms can be classified into high-level and low-level categories. High-level hybrids combine entire metaheuristics without altering their internal mechanisms, creating new solutions. Low-level hybrids modify the internal components of one metaheuristic with elements from another. Additionally, hybrid solutions can be organized as relay or teamwork methods (Talbi, 2002). In relay methods, the output of one metaheuristic becomes the input for the next step. In teamwork methods, multiple algorithms work in parallel to find the best solutions.

2.3.5 Data Processing

Datasets often contain data points of varying magnitudes, which can skew results if not properly processed. One way to mitigate the impact of outliers is to preprocess the data. An outlier is defined as *"an observation that lies an abnormal distance from other values in a random sample from a population"* (NIST, n.d.). To handle outliers, data can be normalized, for instance, using the `QuantileTransformer` from the `sklearn` library in Python (scikit-learn, n.d.-c). This method is non-linear, which could potentially influence the results. Essentially, all values are compared to their normal distribution within the analyzed parameter and then adjusted accordingly.

Another way of preparing the data for clustering is to select the features that will be used. In other words, what variables that will be used to put products into different clusters, for example sales quantity, and product volume. One important consideration is that the features selected for the clustering are illustrating the reality in a good way (Columbia University Mailman School of Public Health, n.d.)(Everitt

et al., 2011). For example if a group of people should be clustered based on postal code, then a bad feature would be hair length.

2.4 Literature Synthesis

After discussing various concepts in warehousing and artificial intelligence, it is useful to summarize the findings. An overview of the articles used can be found in Appendix A. Warehousing fundamentals are straightforward: a warehouse requires a layout with storage racks, incoming SKUs need to be placed in storage, and customer orders must be picked from storage. As previously mentioned, picking operations are notably costly, often representing around 50% of a traditional warehouse's expenses (Rushton et al., 2021). Hence, improving slotting and picking processes can significantly reduce overall costs. Additionally, considering the health of warehouse workers is important, although this aspect is often overlooked in slotting optimization literature despite its significant impact on health and costs.

In the field of AI applied to slotting, there is a notable gap in the literature regarding the use of hybrid algorithms that incorporate ergonomic factors (see Appendix A). Moreover, recognizing that class-based storage strikes a balance between ease of implementation and performance directs the focus of this research. By focusing on clustering to create product classes, the solution can potentially be implemented more easily. Furthermore, integrating clustering into a hybrid algorithm can simplify the optimization of SKU locations by reducing the problem's complexity.

3
Methodology

This chapter presents the methodology adopted for this study. Including an overview of the research design, a description of the quantitative approach, and the steps taken to ensure high-quality research.

3.1 Research Design

For the initial phase of the research, a detailed research strategy was developed, structured according to the steps outlined in "Business Research Methods" by Bell et al. (2019) ensuring a systematic and thorough exploration of the research subject.

The step-by-step framework is presented in Figure 3.1. The first step was to define the research area, providing a clear direction for the study. The next step consisted of a literature search to deepen the understanding of the subject matter. Following this, an in-depth analysis of data collected from the case company was undertaken, which was the data that the generated solution was tested on. Lastly, conclusions were drawn from the generated solution used on the case company data.

A critical aspect of the research design was the iterative loop between data analysis and solution refinement. This iterative process was essential for enhancing the solution, as it involved continuous evaluation and adjustment based on the outcomes produced by the initial solutions. Without this feedback loop, achieving a refined and effective solution would have been challenging (Bell et al., 2019). Also, as mentioned when former articles were presented, other researchers has used a similar method of creating a solution in terms of a program, and then drawing conclusions by providing data to the model.

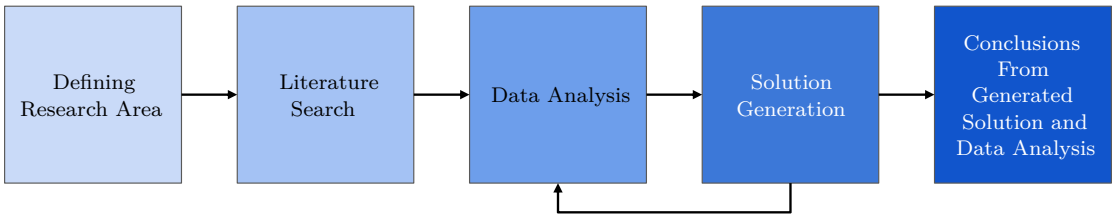

Figure 3.1: The outline of the main steps of the research design.

To outline the timeline for the proposed research steps, an overall timeplan was created (see Figure 3.2). As can be seen in the figure, the steps overlap, indicating that many research activities were conducted in parallel. The most time-consuming

phases were data analysis and solution generation, each taking approximately two months.

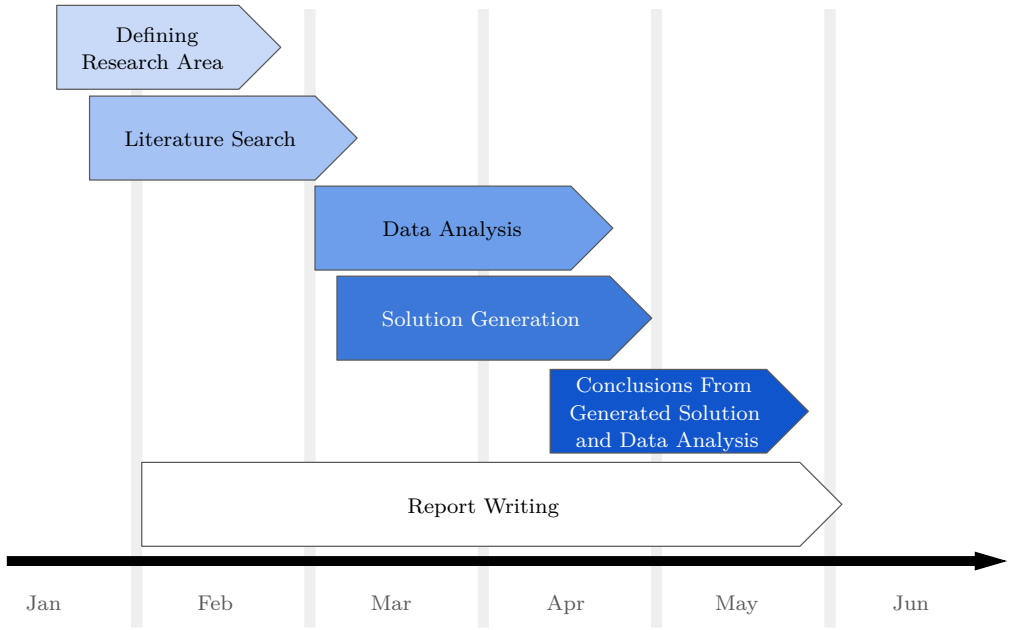

Figure 3.2: Rough time plan for when the proposed steps in the research design were planned to be carried out.

3.2 Quantitative Research

To optimize slotting, a quantitative case study approach was adopted using data from a case company. This method was chosen due to its ability to identify cause-and-effect relationships and to determine correlations between variables (Davidson & Patel, 2019). Since the problem area lies within engineering, dealing with concrete data rather than subjective opinions makes a quantitative approach particularly appropriate.

Another reason for choosing a quantitative method is its applicability to other situations beyond the specific case used in this report (Davidson & Patel, 2019). This aspect is crucial, as the results should be generalizable and not dependent on the specific researcher or the case company. Although using a single case company might limit generalizability, the selected company was deemed representative of the larger population, justifying this approach.

With the quantitative approach, ensuring the validity and reliability of the data was critical. Validity refers to the relevance of the data collected for the research objectives (Davidson & Patel, 2019). This was ensured by reviewing the data after completing the literature review to confirm its alignment with established research parameters. Reliability is about to the accuracy of the data collection process (Davidson & Patel, 2019). To mitigate potential inaccuracies, a preliminary data analysis was conducted, including verifying figures like customer numbers and order

volumes with the data provider.

The verified data formed the basis for developing the AI-driven slotting optimization solution. Using Python libraries, the solution was tested and refined. Feedback from industry experts, such as senior logistics consultants, further enhanced the solution, ensuring it fit its intended environment and provided practical insights into feasible slotting solutions.

A detailed description of the implementation process is provided in the following chapter, Implementation.

3.2.1 Case Company

One aspect to consider when selecting a case company to use for the study was the potential of learning from the specific case (Bell et al., 2019). When selecting a case company, the potential for learning and deriving general conclusions was a key consideration (Bell et al., 2019). As an instrumental case study, the selected company provided insights into the broader topic.

To maintain confidentiality, the case company remained anonymous throughout the report. This commitment to anonymity was crucial for establishing trust and ensuring confidentiality during the research (Davidson & Patel, 2019). The company, an industrial firm with several billion SEK in revenue, has its main production facility in connection to the warehouse used for data collection. The warehouse receives orders three main sources: in-house production, internal sales organizations, and direct customers. This results in a wide variation in order sizes. Internal sales organizations often place large orders to supply entire markets and fill containers to minimize transport costs. On the other hand, direct customer orders are typically smaller, leading to a diverse range of order sizes.

3.3 Ensuring High-Quality Research

Three key criteria used for ensuring that the conducted research is of high quality are reliability, replicability, and validity (Bell et al., 2019).

3.3.1 Reliability

Reliability concerns the consistency of the research findings (Bell et al., 2019). In the context of this study measures have been taken to ensure reliability. For quantitative analysis, using consistent measurements across various scenarios creates results that can be compared to each other. Similarly, for the initial literature search, consistency is sought through a systematic approach to data collection and analysis.

3.3.2 Replicability

Replicability refers to the research being clearly described about how it was conducted (Bell et al., 2019). Doing so enables other researchers to do the same research again to verify the findings, or potentially trying if the results are the same when tested in a different setting. The literature search in this report was conducted in way that it can be made by another researcher if needed, for example by using the same query in Scopus when searching for literature. Also, by using a quantitative case study, a researcher can use the same AI-tools as in this report, but in another setting which also enable replicability.

3.3.3 Validity

Validity in general ensures that the research accurately addresses the research questions (Davidson & Patel, 2019). This is of importance, since low validity would generate results that are not aligned with the questions being asked. Validity can be divided into several subcategories: *measurement validity*, *internal validity*, *external validity*, and *ecological validity* (Bell et al., 2019).

Measurement Validity Measurement validity is about making sure that measurement made are connected to the topic of the research, and not measuring something else (Bell et al., 2019). This was considered in this work by understanding the key measurements with respect to factors such as travel distance and ergonomics. For example, measuring the travel time for a picker in a warehouse does not give the full picture on how the travel distance is affected since the travel time is depending on more factors than the distance.

Internal Validity Internal validity is ensuring that the variations observed in the study results are the direct outcomes of the variables manipulated or studied, rather than being influenced by external, unrelated factors. (Bell et al., 2019). To uphold internal validity, the overall problem setting was understood by, for example, testing different input values and verifying that they created the output that the researchers thought would appear. A concrete example of this was when optimizing the internal order of the products within clusters, then it could be seen that the output in form of ergonomics and picking distance was reduced, which was in line with prior assumptions.

External Validity External validity ensures that the research findings can be generalized to other contexts beyond the specific study (Bell et al., 2019). This aspect is closely related to replicability, as the research should be able to be repeated in various settings to confirm its credibility. Achieving high external validity can be challenging in case studies. In this report, external validity was addressed by using data from multiple warehouse settings, allowing the developed model to be tested in different contexts.

Ecological Validity Ecological validation refers to the extent to which the findings of the study can be applied in a real-world setting (Bell et al., 2019). For the conducted research, ecological validity is achieved through the usage of a real-world case company. This is beneficial due to the possibility to test the generated solution on data collected from reality, rather than testing it on a superficial set of data. Having used a case company for this research, the study aims to produce insights that are not only technically valid but also practically applicable.

4

Implementation

This chapter provides an in-depth explanation of the warehouse optimization program's functionality. It serves both as a detailed overview of the program's functionality and a justification for the decisions made during its development.

4.1 Investigation of the Dataset

To give insight on certain characteristics of dataset, the Apriori algorithm was used. The `mlxtend` library in Python (Raschka, n.d.) was used for this. The support-level was 0.5%, meaning that it extracted all order-combinations that occurred in at least 0.5% of all transactions. Although the entire dataset contained over 10,000 products, the Apriori algorithm presented just over 600 (out of 40,000 total orders) order combinations with the support of 0.5%. The length of the orders were between 1 and 200 products and contained around 1.4% of all individual products. In other words, orders that happen in at least 0.5% of all transactions contain about 1.4% of all available SKUs. A visualization of this insight can be seen in Figure 4.1.

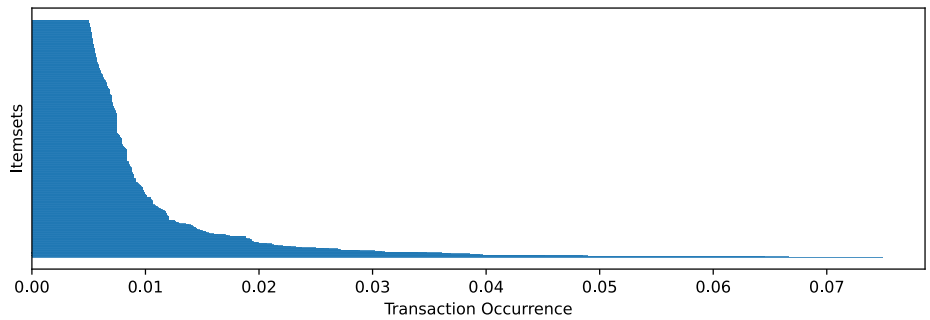

Figure 4.1: Outcome of Apriori algoithm with the limit for the support set at 0.5% and plotted for orders with a length of between 1 and 200 products.

Another insight to the data is that it contained over 200,000 order lines, which in turn create the over 40,000 orders. To reflect on the number of order types that are occurring in at least 0.5% of all orders, it is worth noting that there are roughly 18,000 unique orders. The popular orders found by the Apriori algorithm represent just over 3% of all the order combinations present in the data. Regarding order length, the longest order is just over 200 items long. Worth mentioning is that the order combination with the top 5 most occurring orders are of single items.

Seeing these results made the development of the final program not consider any correlation between products and what kind of orders they occur in, since so few products were represented within these common orders. This reduced complexity when clustering the products. It was deemed reasonable since the order data was considered representative for other cases as well, meaning that any correlation on order-level would potentially not yield a benefit worth pursuing in this implementation.

4.2 Overview of Algorithm and Input Data

The created algorithm consists of two primary parts, also referred to as modules. Firstly, the program executes clustering of the demand data (see Figure 4.2), dividing the products into several clusters, using the K-Medoids algorithm. This stage only require order line data, which will be described later in this section. Lastly, the program makes use of a warehouse layout and takes the generated clusters and runs a genetic algorithm, firstly for optimizing the placement of entire clusters, then it performs another optimization for the placement of the products within each cluster.

With a modular design, the clustering can be executed in isolation, which is beneficial when not having a defined warehouse layout. The optimization itself have the possibility of optimizing with respect to two parameters, namely the ergonomics and the picking distance. Also, a third alternative is available in form of a combined optimization between the two previous parameters. The structure has taken inspiration from the one provided by Li et al. (2016) and Yang and Nguyen (2016).

Worth mentioning are the three outputs from the algorithm: two from the clustering module and one from the product placement module. The clustering module generates cluster characteristics, which have the centroids of each cluster presented. In addition, a list of all products found in the demand data are presented with corresponding characteristics as well as the cluster to which every product belong. Output from the product placement module is a product list similar to the one from the clustering module, but in this stage, it also contains the placement for each product within the warehouse.

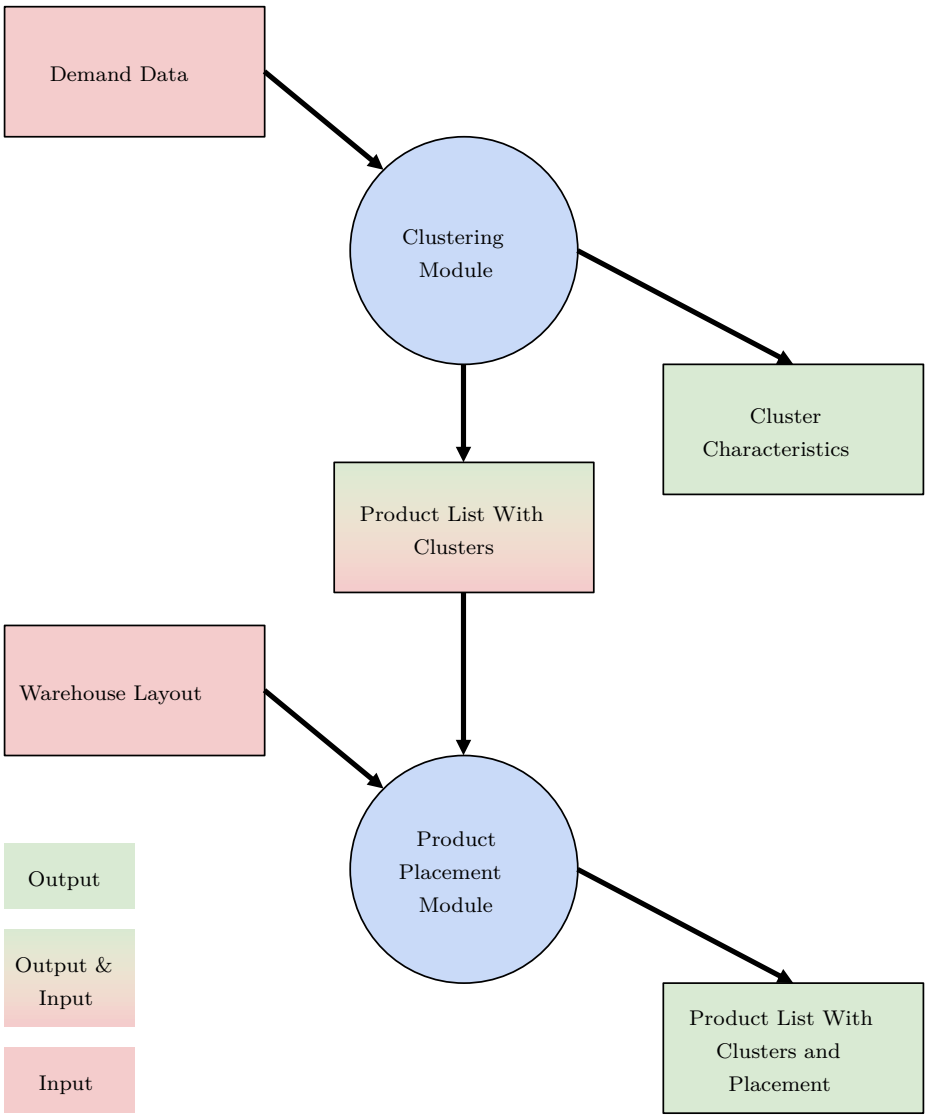

Figure 4.2: Visualization of the structure used in the created algorithm for clustering and optimization, with the two modules represented by blue forms.

To provide insight to the data that has been used as input for the two modules, three examples of data will be provided for the reader to grasp the detail needed for executing the algorithm. The first data to be presented is the demand data, which is needed for the clustering module. It contains several parameters/features on order line-level (see Table 4.1), which contains data for each pick made in the order. Each order line represents a single product, which may have a quantity greater than one. Consequently, an order containing multiple SKUs will have several order lines. This data format is commonly used by the case company (and other companies), which justifies its inclusion and the subsequent data aggregation within the clustering module.

Table 4.1: Generic structure of demand data used for aggregation on product-level (data is fictional, but with the same structure as the case data).

Order Line	Order Number	Part Number	Sales Quantity	Order Line Weight
1	A1	A	5	15
2	A1	B	3	3
3	A1	C	1	10
4	A2	X	5	14
5	A2	A	7	21
6	A2	Y	1	1.5

Before clustering the data, it is aggregated on product-level. The aggregation is made on product-level to enable clustering of products and not orders. By using the data from the order lines, it is possible to determine the standard deviation of the sales quantity for each unique product based on the quantity in each order line where the product is present. This adds a variable to the data which is used as input for the clustering analysis. Picking frequency is determined by how many order lines each product appears in. A generic presentation of the processed data used for the clustering is presented in Table 4.2.

The reason for selecting these three variables (total order line weight, standard deviation of sales quantity, and pick frequency) is due to their ability to describe the reality in a good way for the clustering. Given the goal of an optimized solution for picking distance and ergonomics, the total weight picked can be related to the energy expenditure for picking the products, the standard deviation describe how the demand fluctuates, and the pick frequency is an indicator for how close products should be to reduce picking distance if they are picked often.

Table 4.2: Generic structure of aggregated data used for the clustering algorithm (data is fictional).

Part Number	Total Order Line Weight	Std of Sales Quantity	Pick Frequency
A	90	1	15
B	45	1.5	30
C	350	2.5	23
X	154	3	5
Y	27	0.4	7

Another dataset needed is the layout of the warehouse, working as an input for the product placement module. Without a layout, it is impossible to optimize any sort of placement. A simple approach was taken, which is an Excel-file with numbers representing shelves and the floor as well as text for different functions within the warehouse. The number 0 represent the floor, any integer greater than 0 represent a shelf, with the number on it stating the height of the shelf with respect to the number of storage locations in the vertical dimension. The text ”x” represents walls, and the

text ”start” and ”end” are the codes for the starting- and ending points respectively of the the picking route which is key information when optimizing within the product placement module. In Table 4.3 an example of a warehouse layout is presented.

Table 4.3: Example of data for warehouse layout with ”x” representing walls, ”0” the floor, integers greater than 0 the number of storage spots in vertical direction, and ”start” and ”end” the start and end points of picking rounds.

x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
x	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	end	x
x	0	5	5	0	6	6	0	6	6	0	4	4	0	8	8	0	3	0	x
x	0	5	5	0	6	6	0	6	6	0	4	4	0	8	8	0	3	0	x
x	0	5	5	0	6	6	0	6	6	0	4	4	0	8	8	0	3	0	x
x	0	5	5	0	6	6	0	6	6	0	4	4	0	8	8	0	3	0	x
x	0	5	5	0	6	6	0	6	6	0	4	4	0	8	8	0	3	0	x
x	start	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x
x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

4.3 Clustering Algorithm

As outlined in chapter 4.2, the first part of the algorithm consisted of a clustering algorithm. This section describes the systematic approach taken for clustering including the steps for evaluating different clustering methods and determining the optimal number of clusters for the dataset. For the clustering, `scikit` library in Python was used (scikit-learn, n.d.-a, n.d.-b; scikit-learn-extra, n.d.).

4.3.1 Cluster Analysis

The methodology for the cluster analysis was inspired by the steps outlined in the book by Everitt et al. (2011). The steps are presented below.

(i) Objects to cluster: Selecting the objects to cluster was the first part of the clustering algorithm. Since products were to be slotted in the warehouse, products were selected as the objects to cluster. The products (or SKUs) were found in the demand data. The number of products used in the clustering depended on the number of available storage spots in the warehouse, since this number must be equal or greater than the number of products to be slotted.

(ii) Variables to be used: A large part of the clustering was to define variables to use. Irrelevant variables were excluded, and variables should only be included if there is a good reason that they will define the clusters (Everitt et al., 2011). This mindset was used to evaluate the features systematically. This process resulted in three features as described in section 4.2.

(iii) Missing values: As the data did not have a lot of missing values a specific plan for handling missing values were not needed. The few missing values found in the dataset was converted from NaN to 0.

(iv) Clustering method: The choice of method should be based on types of clusters suspected and insensitivity to error (Everitt et al., 2011). The methods that were evaluated for their stability are described in chapter 2.3.1. The different method's stability was evaluated by examining how consistently each data point was assigned to the same cluster across different iterations for different number of clusters.

This most common assignment is called the "mode", which refers to the process of identifying the most frequently assigned label for each data point (Hastie et al., 2009). The stability score of a particular method was calculated by dividing the count of the mode by the total number of runs. A higher stability score indicates that data points are more consistently assigned to the same cluster, suggesting that the clustering results are reliable and reproducible under the same conditions.

Worth mentioning is that the clustering analysis was made on scaled data, using a standard distribution function presented in Chapter 2.3.5. This made outliers not impact the clustering to a greater extent.

(v) Number of clusters: The decision on the number of clusters, K, to be used was primarily guided by the specific goals of the analysis, as highlighted in *The Elements of Statistical Learning* (Hastie et al., 2009). Similar considerations are noted in another study (Nafar, 2022) which selected cluster number based on warehouse layout and practical constraints. Therefore, we opted to choose the number of clusters manually. However, to support this manual selection, we still evaluated various cluster numbers using cluster validation metrics that do not rely on class labels, detailed in Section 2.3.1. This was done by computing the Silhouette score, Davies-Bouldin score and Calinski-Harabasz score for different methods and different cluster numbers, ranging from 2 to 100.

(vi) Replication and testing: For replication and testing, different numbers of products included were tested with results performing in similar manners. Also, another data set provided was used as a verification, but this data was not allowed to be presented in the report due to confidentiality. This other data set yielded similar outcomes as for the main one used in the thesis.

(vii) Interpretation: To understand the generated clusters, plotting was the main method used. This created a graphical representation to grasp how the clusters differ between each other and how the potential overlap between them looks like. When presenting the clusters in a plot, the scaled data was used to easier

understand and interpret the clusters that had been created.

4.3.2 ABC Analysis

To compare clustering methods with traditional clustering, an ABC analysis was done on the dataset. It was made in the way described in the theory, considering the picking frequency as the variable for analysis and the 80%-limit for A-items and a 95%-limit for B-articles.

4.4 Goal Functions

Running the optimization required goal functions (also known as fitness functions). Three functions were created, one for each type of optimization (ergonomics, picking distance, and the combination of both). The goal functions enables the program to evaluate a given solution of slotted products with respect to the three optimization variants.

4.4.1 Ergonomics

For the ergonomics optimization, the function considers the vertical height from the floor to the product in question and the total weight of the demand for that specific product. Three different scores are decided between for the height. If the product is at the lowest level, or at a level higher than the third, the highest score is given to the *ergonomic factor* (see Figure 4.3), since it means that the product is picked from the ground because the higher-level products are brought down to the floor and lifted from there. The second and third level receive better scores due its better ergonomics compared to the lowest and highest levels. The third level gets the best score, while the second shelf level gets the second-best score. This is based on the research by Marras et al. (1999), Calzavara et al. (2017), and Waters et al. (1993), although this modelling is a vast simplification of their results.

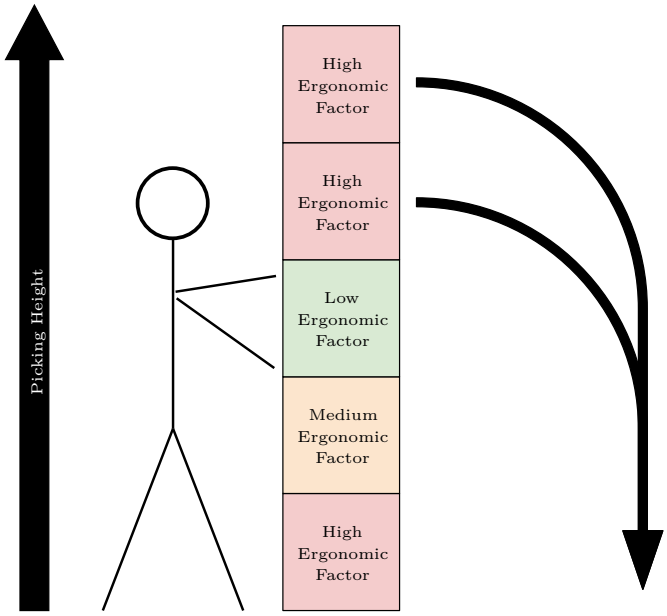

Figure 4.3: Visualization of how the ergonomic factors are determined based on the picking height.

Having an ergonomic factor for each product allows for the calculation of an *ergonomic score* for each product. This is based on the research by Calzavara et al. (2017) which propose an energy expenditure calculation when evaluating ergonomics. In this application, the force in the work-equation corresponds to the total weight of the product demand (see Equation 4.1). The distance is represented by the ergonomic factor. This results in an equation that account for products that are heavy and lifted more seldom and products of lower weight but with higher demand, in other words the energy needed.

$$\textit{Ergonomic Score} = \textit{Total Weight of Demand} \cdot \textit{Ergonomic Factor} \tag{4.1}$$

This results in a goal function for the ergonomics (see Equation 4.2, where k is the number of products) that aims to minimize the total ergonomic score for the entire line of products. This is achieved by placing products with high total weight of the demand in storage locations with low ergonomic factors and products with low demand in weight in spots with high ergonomic scores. For workers, this has the potential of reducing the physical stress on them.

$$\textit{Goal Function}_{Ergonomic} = \sum_{n=1}^{k} \textit{Ergonomic Score}_{n} \tag{4.2}$$

4.4.2 Picking Distance

The second goal function focus on measuring the picking distance. For this, a simple approach is used, not considering advanced routing heuristics. The distance between the picking round starting point and the products is calculated by the sum of the absolute values in x, y, and z-direction (see Equation 4.3). These routes are doable

in the real-life scenario, and represent a middle ground between the S-shape and return method heuristic (de Koster et al., 2007).

$$Distance_{Product} = |x_{Product} - x_{Start}| + |y_{Product} - y_{Start}| + |z_{Product} - z_{Start}| \quad (4.3)$$

To calculate the picking distance for an order, the distances for all products in the order are summed and the distance between the picking route start- and end point is added to that sum (see Equation 4.4 and 4.5 where i is the number of products in the order).

$$Distance_{Start-End} = |x_{Start} - x_{End}| + |y_{Start} - y_{End}| + |z_{Start} - z_{End}| \quad (4.4)$$

$$Distance_{Order} = Distance_{Start-End} + \sum_{n=1}^{i} Distance_{Product_n} \quad (4.5)$$

The order distance is the basis of the goal function for picking distance. The function focuses on minimizing the total picking distance considering all orders (see Equation 4.6 where k is the number of orders). Minimizing this function puts products that have a high picking frequency close to the starting point of the picking rounds, therefore reducing the total distance for the orders in the given dataset. If using, for example, the return method, similar patterns could have been generated since placing product close to the main aisle or the starting point reduce the picking round distance.

$$Goal\ Function_{Distance} = \sum_{n=1}^{k} Distance_{Order_n} \quad (4.6)$$

4.4.3 Combination

Finally, a combined goal function addresses the case of a trade-off between picking distance and worker ergonomics. It incorporates the two goal functions (ergonomics and distance) and applies a weight to each function for adjusting te focus on each of the two variables. The higher the weight, the greater the focus on that variable when optimizing. Equation 4.7 shows the combined goal function, where w_E and w_{PD} are the weights for the ergonomic and picking distance goal functions, respectively.

$$Goal\ Function_{Combined} = w_E \cdot Goal\ Function_{Ergonomic} + w_{PD} \cdot Goal\ Function_{Distance} \quad (4.7)$$

In conclusion, three goal functions are used to minimize the ergonomic strain on the pickers and the picking distance for the orders using a simple routing calculation. To create a trade-off optimization between ergonomics and distance, there is another goal function to cover that case.

4.5 Logic of Placing Products Within the Warehouse

Enabling optimization of product placement within the warehouse required development of a logic of placing out products to different storage locations. This logic is used when evaluating how good an optimization performs, in other words, this logic enables the goal functions to be evaluated. Three main areas of consideration have been made with regards to placing out the products in the warehouse: how storage locations are numbered, how the location of products are determined, and how free space is distributed between the clusters.

Starting with the shelf numbering, several aspects had to be taken into consideration. Since clusters should not be split between different areas in the warehouse, the numbering must be so that, for example, storage location 1 to 100 must be in a logically coherent way. To describe this numbering further, consider a warehouse in three dimensions (X, Y, and Z), which is represented in Figure 4.4. Numbering of the storage locations is crucial in this application, since it determines how the clusters are located in the warehouse.

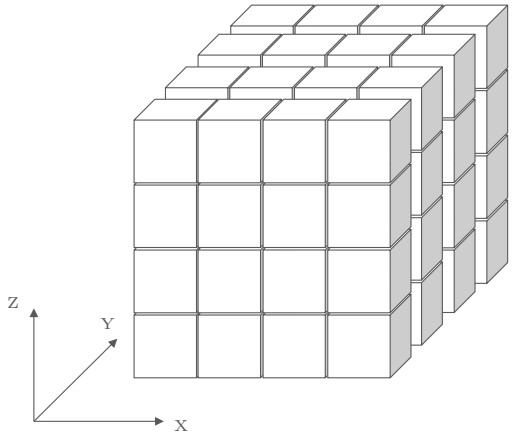

Figure 4.4: Generic warehouse in three dimensions.

One consideration was whether the clusters should be ”standing” (see Figure 4.5) or ”laying” (see Figure 4.6). It was decided to use ”laying” clusters since that typically result in the same Z-coordinate for all products in that cluster. This is important, since it enables optimization of ergonomics on cluster level, which is determined by the height in Z-direction. If clusters would have been ”standing”, the ergonomic scores would have been difficult to impact when considering cluster-level optimization, as the initial product placement being based on product number within the clusters.

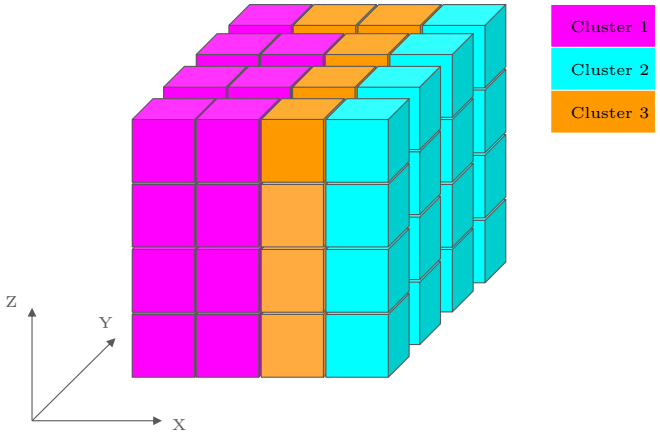

Figure 4.5: Example of the clusters ”standing up”.

In the code, the ”laying” cluster positioning was created by numbering the storage locations by having the Z-direction as the outer loop, the X-direction as the middle loop, and the the Y-direction as the inner loop. Also, to not split clusters when the X- and Y-direction are reset in the loops, the program sees if the loop is at the end for any of the two variables. If so, the loop run backwards the next time. This ensures that clusters are ”laying down” when placed based on the storage location numbering as presented in Figure 4.6.

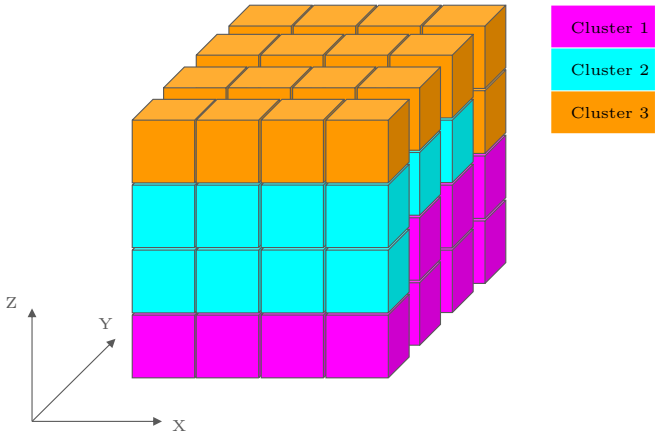

Figure 4.6: Example of the clusters ”laying down”.

Implementing this way of numbering storage location yields a numbering that has the highest storage numbers on the top of the warehouse, as illustrated in Figure 4.7. This figure use the described way of numbering the slots.

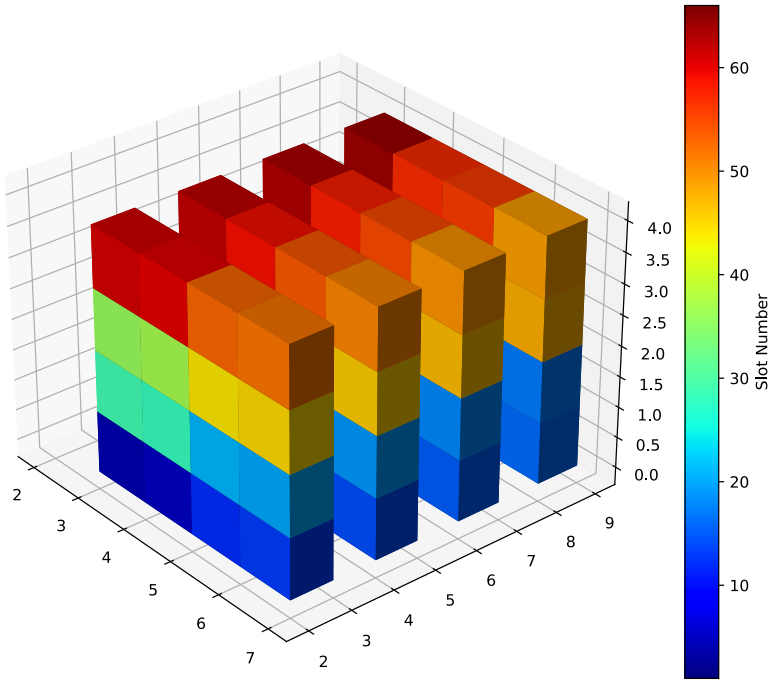

Figure 4.7: Visualization of implementing the selected way of number the storage locations.

Moving on to the placement of products in the warehouse layout highlights another area of consideration in the logic of the program. A function was developed to place the products in the warehouse. It takes the information of all products and their corresponding clusters, the number of available storage locations, and the order of which the clusters should be placed out in the warehouse. This function then allocates the first cluster in the cluster-order list to the first locations in the warehouse based on the numbering on the shelves. This is then continued throughout all available products. To exemplify this process, consider a set of 10 products with corresponding products (see Table 4.4), and a warehouse with 10 available storage locations (numbered from 1 to 10).

Table 4.4: Example of products with corresponding clusters.

Part Number	Cluster
2	2
1	3
3	2
4	4
5	4
10	1
7	3
8	1
9	1
6	2

Furthermore, assume that these clusters should be placed in the order of $[4, 2, 3, 1]$, which could be a combination that the genetic algorithm wants to test. The described function will then take the exact order of the products in the list of products and their clusters as the internal order within the clusters. The output in the example case would then be as presented in Table 4.5.

Table 4.5: Example of products with corresponding clusters and storage locations exemplifying how the program works when placing out products.

Part Number	Cluster	Storage Location
4	4	1
5	4	2
2	2	3
3	2	4
6	2	5
1	3	6
7	3	7
10	1	8
8	1	9
9	1	10

This functionality enables the genetic algorithm to modify the order of the clusters to assess which combination that performs the best. In addition, it is also possible to rearrange the order of the products within the clusters by changing the input before it is used in this function. The change needed is the order of which the products are in the product list, since that is the order of which the products are located within the cluster. Effectively, this enables the genetic algorithm to also test different orders of the products within the clusters to find more optimal (and detailed) solutions. The latter part enables the genetic algorithm to optimize on product-level.

In practice, it can be difficult to have a warehouse with the exact number of storage locations as the number of products. This leads to the final area of consideration when placing the products in the warehouse, namely, how to handle the situation of more storage spots than the number of products. The approach used in this

program is to have a split (that the user can decide) between free storage between each cluster and at the end of all clusters, resulting in free locations on the top of the warehouse structure. For instance, using the same example case as above, but with 15 available storage locations, and wanting to have the split of 60% free spots between the clusters and 40% after the clusters, it results in a storage assignment like the one presented in Table 4.6.

Table 4.6: Example of products with corresponding clusters and storage locations exemplifying how the program works when placing out products with 60% of free locations placed between the clusters.

Part Number	**Cluster**	**Storage Location**
4	4	1
5	4	2
Free Slot	*Free Slot*	3
2	2	4
3	2	5
6	2	6
Free Slot	*Free Slot*	7
1	3	8
7	3	9
Free Slot	*Free Slot*	10
10	1	11
8	1	12
9	1	13
Free Slot	*Free Slot*	14
Free Slot	*Free Slot*	15

These three components (shelf numbering, product placement, and free space between clusters) enables the program to consider the reality of a warehouse.

4.6 Optimization Algorithm

Optimizing the article placement in the warehouse was done using a genetic algorithm, implemented with the `DEAP` library in Python (Raschka, n.d.). The parts that were customized were the goal function and the crossover function. Specifically, the crossover function was made to work for the purpose of optimizing the order of the clusters in the warehouse. It is of interest to notice that the optimization can optimize in two ”levels”. The higher level is the location of entire clusters, and when that is done, the program can optimize on a lower level, meaning the arrangement of the products within the clusters.

4.6.1 Crossover Function

A modified crossover function was needed since the input for the product placement function use the order of the clusters place them in the warehouse. Without modification, the crossover function would have created duplicates of clusters, resulting

in not being able to use the placement function. The modified crossover function works as the following: when a crossover point has been determined, the function exchange the order of all clusters that are not already in the individual. This means that the order of the clusters in another individual are used in the individual that it is being paired with (see Figure 4.8). Looking at Figure 4.8 illustrates the characteristics being exchanged is the order of the clusters, rather than a direct exchange. This type of crossover function is used for the overall cluster optimization, as well as for the internal order of the products within the clusters (but then the order of the products within a cluster is the input).

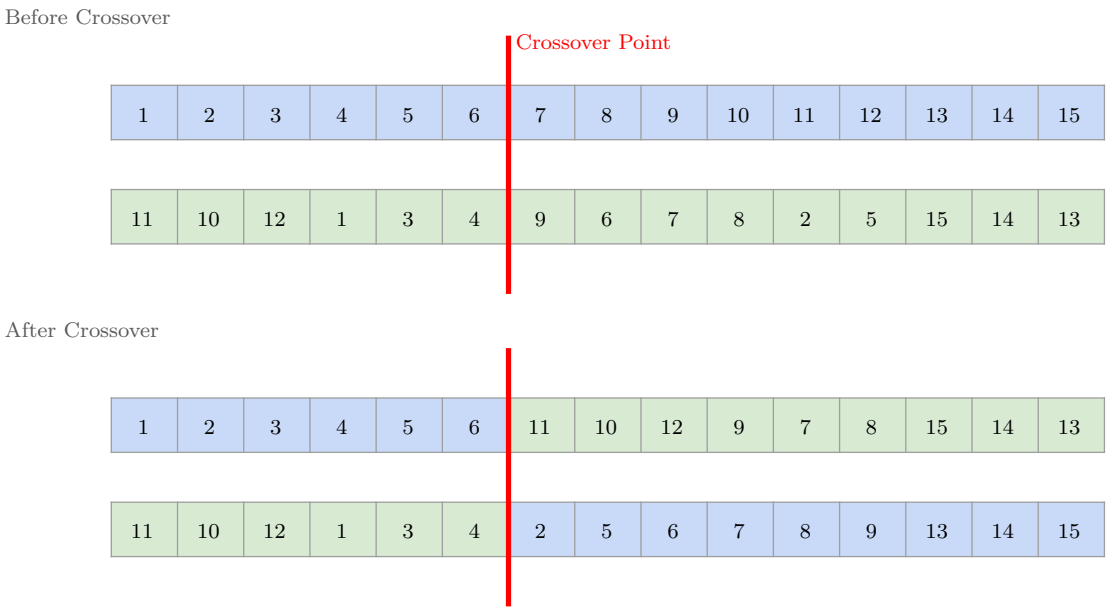

Figure 4.8: Presentation of how the custom crossover function works for two individuals.

4.6.2 Optimization Variants

Two levels of optimization can be used, namely on cluster- and product-level. This is made to reduce the risk of random products affecting the optimization. For example, if a frequently picked product is placed far away from the starting point of the picking rounds, the cluster will be affected of that, although that product might be one of only a few that are frequently picked within that cluster. During optimization of entire clusters, the distance used for each product is measured from the center point of the cluster, rather than the placement of individual products. On the other hand, when optimizing the placement within a cluster, the actual placement of the individuals is used, otherwise the re-order of products would not generate any impact on the distance for the solution. Figure 4.9 illustrates the structure behind the different ways of optimizing the clusters as well as optimization of the internal product order are presented.

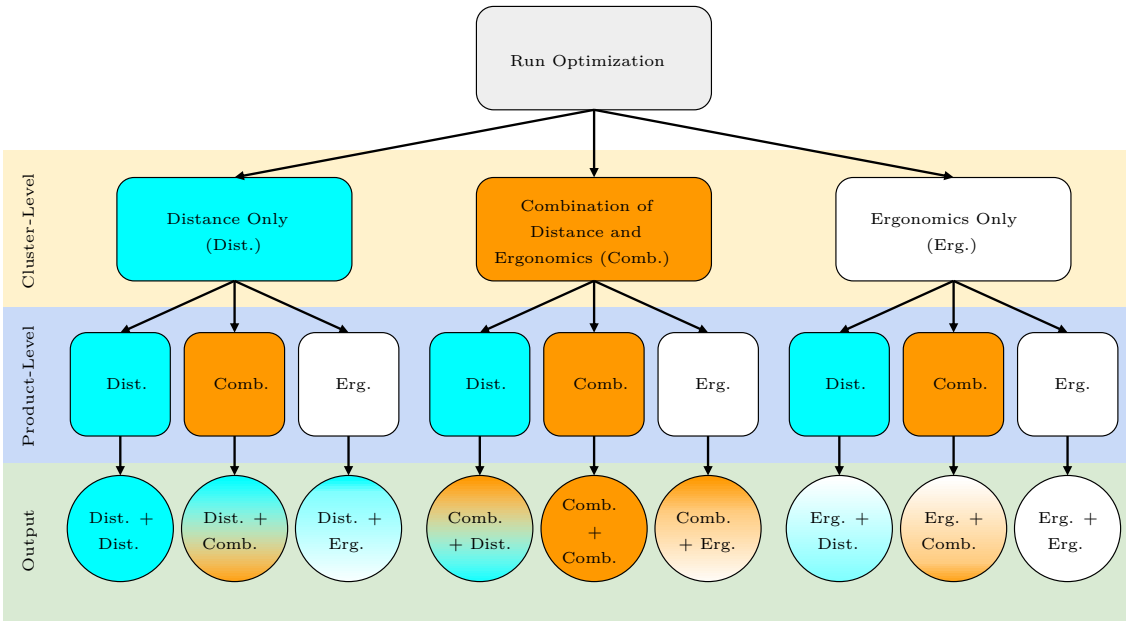

Figure 4.9: All available variants of optimizing the final solution.

5
Results

This chapter reveals the study's results. The chapter begins by presenting the findings from the clustering algorithm, covering both machine learning techniques and traditional methods like ABC analysis. Following this, the results obtained from the optimization algorithm are shown.

5.1 Result from Clustering Algorithm

As outlined in chapter 4.2, the first part of the algorithm consisted of a clustering algorithm. This section describes the results from evaluating different clustering methods, cluster numbers and three different clustering scenarios. Since the used warehouse layout had 1,540 storage locations (see Figure 5.8), the number of products used in the clustering and optimization were 1,400 to enable some free space between the clusters.

5.1.1 Evaluating Method and Cluster Numbers

The result from evaluating different clustering methods is presented in Figure 5.1, both K-Medoids and Agglomerative clustering achieved an average consistency rate of 100%, whereas K-means showed a notably lower consistency at 54.25%. The red lines extending from the blue dots represent the variability in the stability measure. The absence of error bars for K-Medoids and Agglomerative clustering suggests there was no variability in its stability score, which aligns with the average frequency of 100%. A longer line indicates more variability, meaning there is less consistency in how data points are grouped by the clustering method across different runs, which K-means shows in this case. Consequently, K-means was not chosen as a method for clustering.

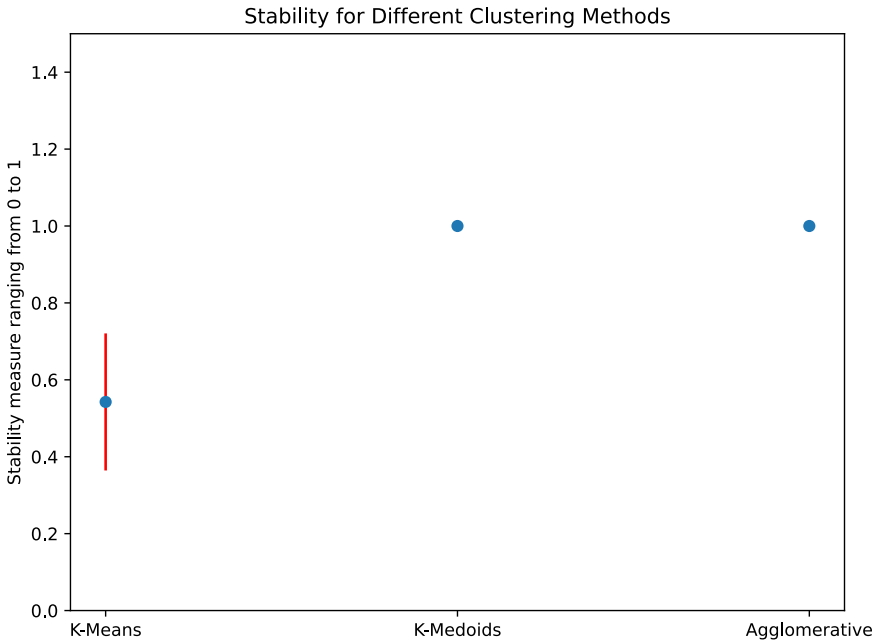

Figure 5.1: Visualization of the stability for different clustering methods.

The results for the Silhouette score, Davies-Bouldin score, and Calinski-Harabasz score, computed for K-Medoids and Agglomerative clustering with varying cluster numbers, are presented in Figures 5.2, 5.3, and 5.4, respectively.

These plots illustrate that having two clusters yields better scores and that fewer clusters generally perform better than many. It could be argued that the data may not be well-separated. However, the primary objective of clustering here is to group data rather than to determine the optimal number of clusters. This analysis was conducted to provide an overview, and aside from the two-cluster scenario, no specific cluster number stands out.

In Figure 5.3, the difference in outcomes between K-Medoids and Agglomerative clustering is the smallest among the three measurements. However, the plots in Figure 5.2 and 5.4 show that K-Mediods performs better than Agglomerative clustering. Therefore, K-Medoids was chosen as the final clustering algorithm for this study due to its superior in overall performance and in maintaining consistent clustering outcomes.

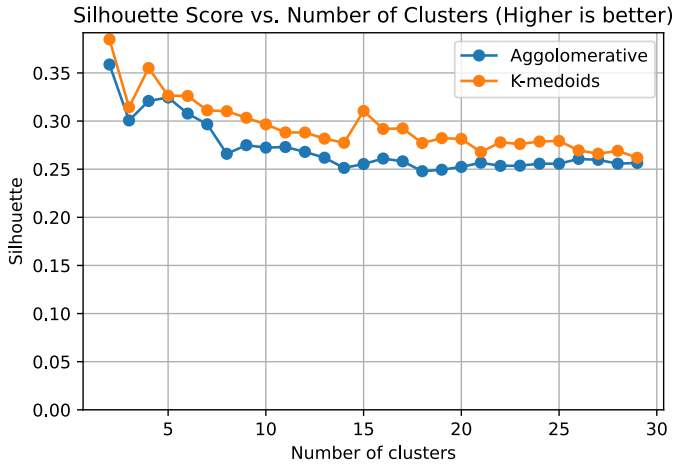

Figure 5.2: Silhouette score for different number of clusters.

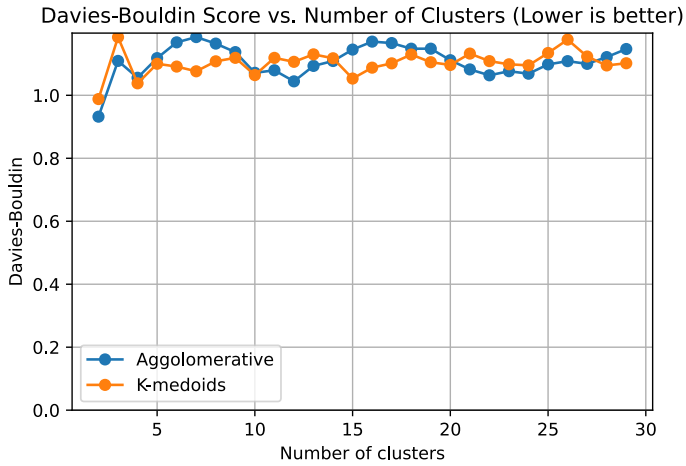

Figure 5.3: Davies-Bouldin score for different number of clusters.

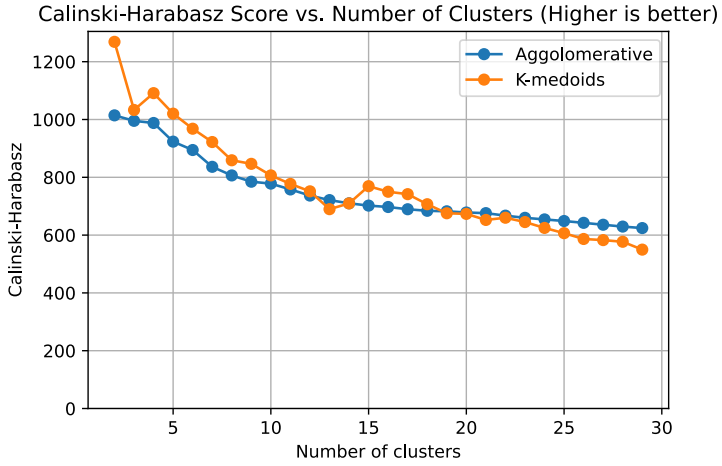

Figure 5.4: Calinski-Harabasz score for different number of clusters.

5.1.2 Three Clustering Scenarios

The clustering algorithm was applied using the K-Medoids algorithm to segment the dataset into different clusters. Three different scenarios were explored, each with a different number of clusters, namely 25 clusters, 3 clusters, and ABC clusters (3 clusters).

The results from using K-Mediods in combination with 25 clusters can be seen in Figure 5.5. The clusters are visually represented in a 3D projection. Clusters are separated from each other, although in middle of the plot, there are more overlap than in the outskirts. Also, the number of products in the clusters are of similar sizes.

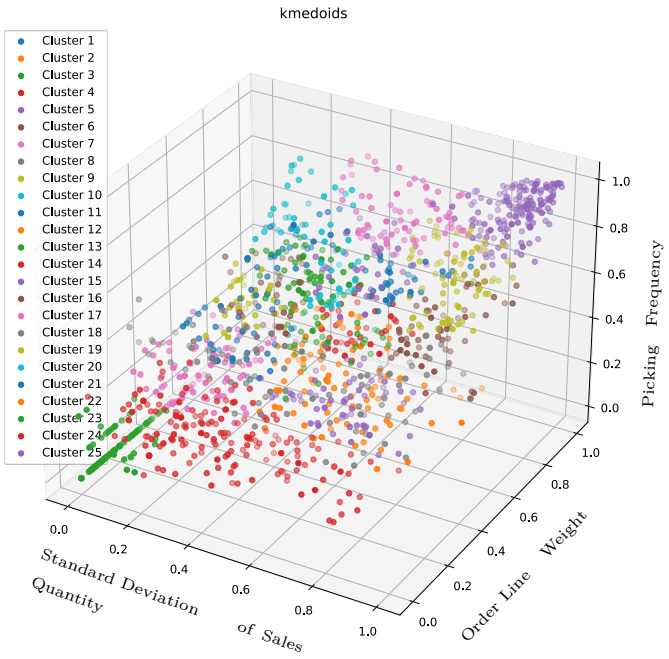

Figure 5.5: Plot of outcome when clustering products with K-Medoids using 25 clusters.

Similarly, Figure 5.6 illustrates the clustering outcome using K-Medoids with three clusters. Similar to Figure 5.5, this clustering also has separated clusters and they are of similar size.

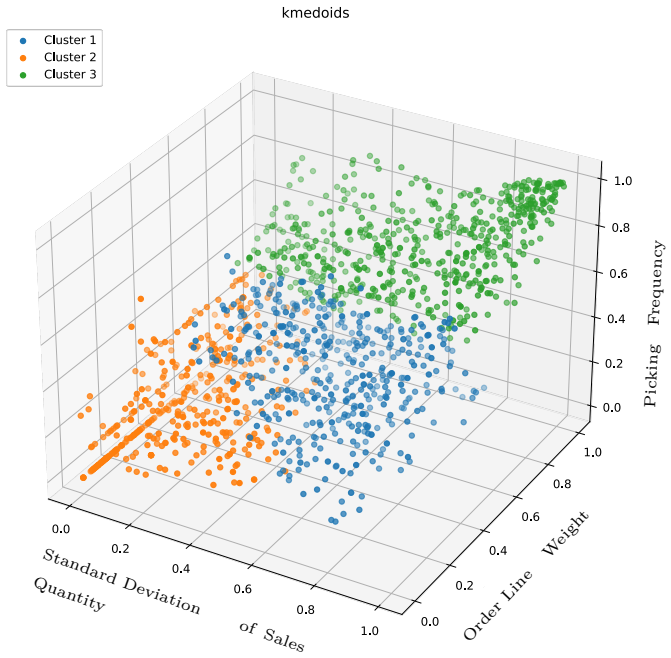

Figure 5.6: Plot of outcome when clustering products with K-Medoids using 3 clusters.

On the other hand, Figure 5.7 presents the results from using ABC-Analysis for clustering, highlighting differences in clustering outcomes compared to the K-Medoids approach. One difference is the size of clusters, which is expected due to the nature of the ABC-clustering method. Another difference is the greater overlap between the clusters for the ABC-analysis, especially when compared to the K-Medoids clustering using 3 clusters.

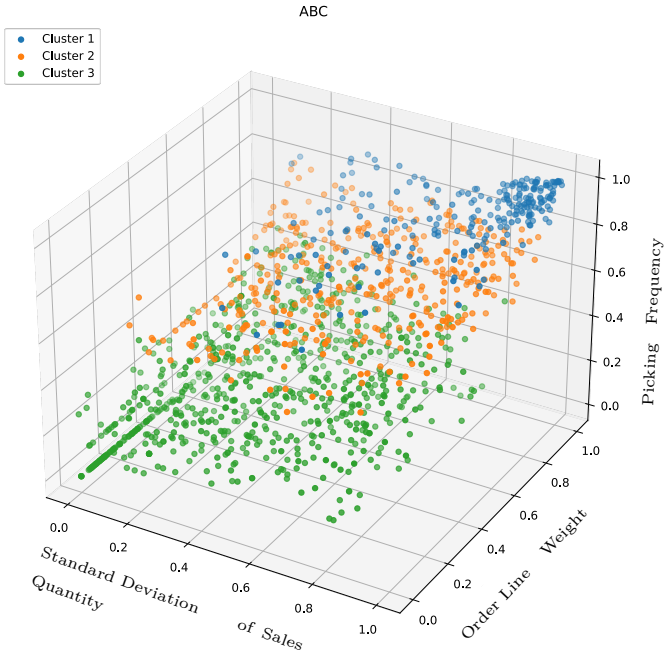

Figure 5.7: Plot of outcome when clustering products using ABC clusters (Cluster 1 is the A-products, Cluster 2 the B-products, and Cluster 3 the C-products).

In summary, the application of K-Medoids for clustering, as illustrated in Figures 5.5 and 5.6, resulted in well-separated and similarly sized clusters, with the best separation observed in the 3-cluster scenario. However, Figure 5.7 shows that the ABC clusters lack clear separation, exhibiting overlap and uneven distribution in cluster sizes. These three clustering scenarios were subsequently used in the optimization process, which will be discussed in the following section.

5.2 Results from Optimization Algorithm

The results of the optimization algorithm are presented using the three previously mentioned clustering scenarios applied to a fictional warehouse layout (see Figure 5.8). The most significant differences in distance and ergonomic scores between optimizing cluster positioning and internal product order were observed when fewer clusters were used. Generally, optimizing the internal order of products improved both ergonomics and picking distance. To compare the different clustering methods, similar parameters were applied across all three scenarios. The optimization runs were selected based on the hierarchy presented in Figure 4.9, resulting in a total of nine different optimization combinations.

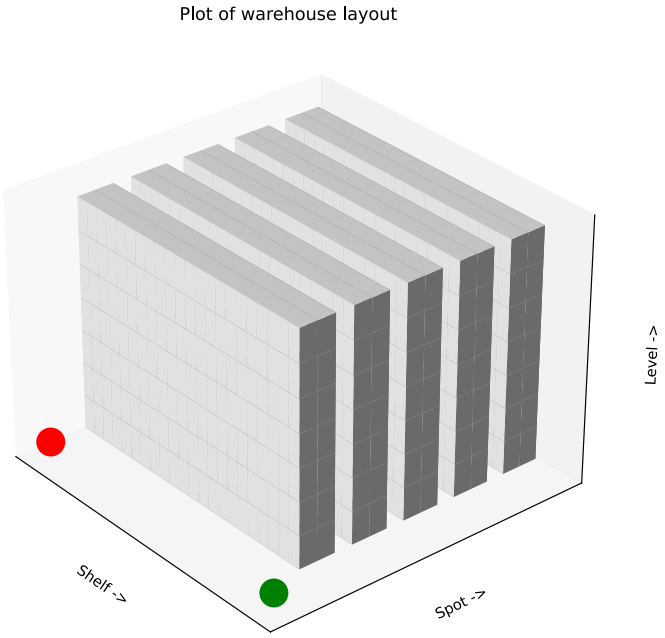

Figure 5.8: Plot of the fictional warehouse layout used for optimization, with the green point representing the starting point of the picking rounds and the red point indicating the ending location.

5.2.1 25 Clusters Using K-Medoids

The first case to be presented is the one where 25 clusters were generated through the K-Medoids clustering. In the first step of only optimizing on cluster level (see Figure 5.9), the distance score is the highest when the ergonomic score is the lowest, but not vice versa. For this run, the combined solution actually got a slightly lower distance score than the purely distance-optimized solution, while at the same time having a significantly lower ($\approx 25\%$) ergonomic score.

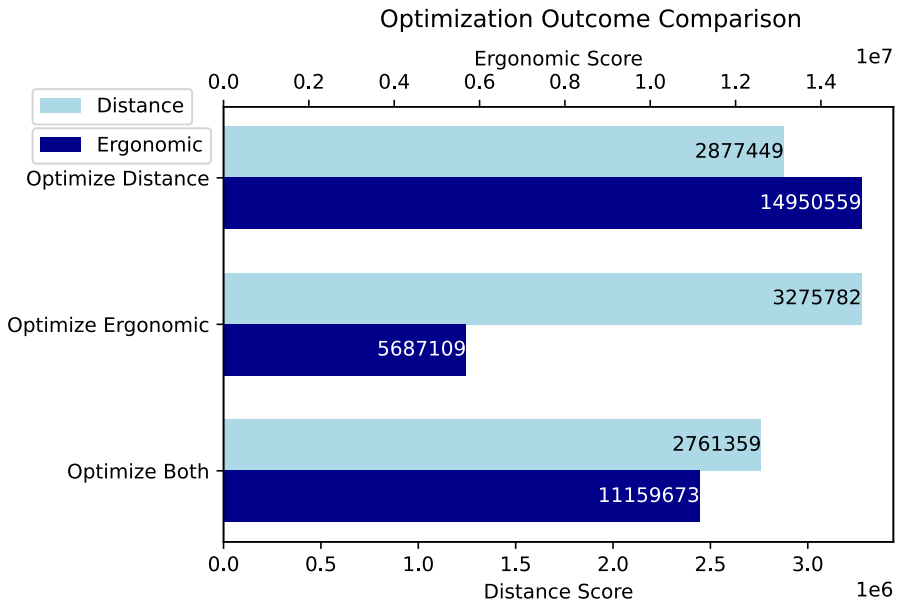

Figure 5.9: Plot of original optimization values when using many clusters.

In Table 5.1, the parameters used for this optimization are presented. It can be seen that for the combined optimization, the distance got a higher weight than the ergonomics, but it still resulted in an outcome that was not like the distance-optimized solution.

Table 5.1: Input variables for optimizing using 25 clusters.

Number of Clusters	25
Population Size	20
Number of Generations	10
Distance Share	0.9
Ergonomics Share	0.1

Moving on to the optimization within the clusters, on product-level (as presented in Table 5.2), the ergonomic score tended to be impacted to a greater extent than distance score in relative terms. Notably, the best-performing product-level optimization were either the combined optimization or the ergonomics optimization. For all the following cases, the best combined solution was determined by the lowest sum of ergonomic and distance score of the combined solutions.

Table 5.2: Best scores after optimizing product order compared to initial cluster positioning for each optimization goal when optimizing using many clusters.

Optimization Goal (Cluster-Level)	Optimization Goal (Product-Level)	Distance Score	Diff. vs Cluster-Level (Distance)	Ergonomic Score	Diff. vs Cluster-Level (Ergonomic)
Best Distance	*Best Combined*	2,809,771	-2.4%	12,032,450	-19.5%
Best Ergonomics	*Best Ergonomics*	3,244,225	-1.0%	6,935,899	+22.0%
Best Combined	*Best Ergonomics*	2,805,295	+1.6%	8,064,181	-27.7%

To understand how clusters performed in relation to picking frequency, the optimization for picking distance (at both cluster and product levels) was plotted in

Figure 5.10. The plot shows that clusters with higher median picking frequencies are positioned closer to the starting point of the picking rounds, indicating that the optimization was effective and reasonable.

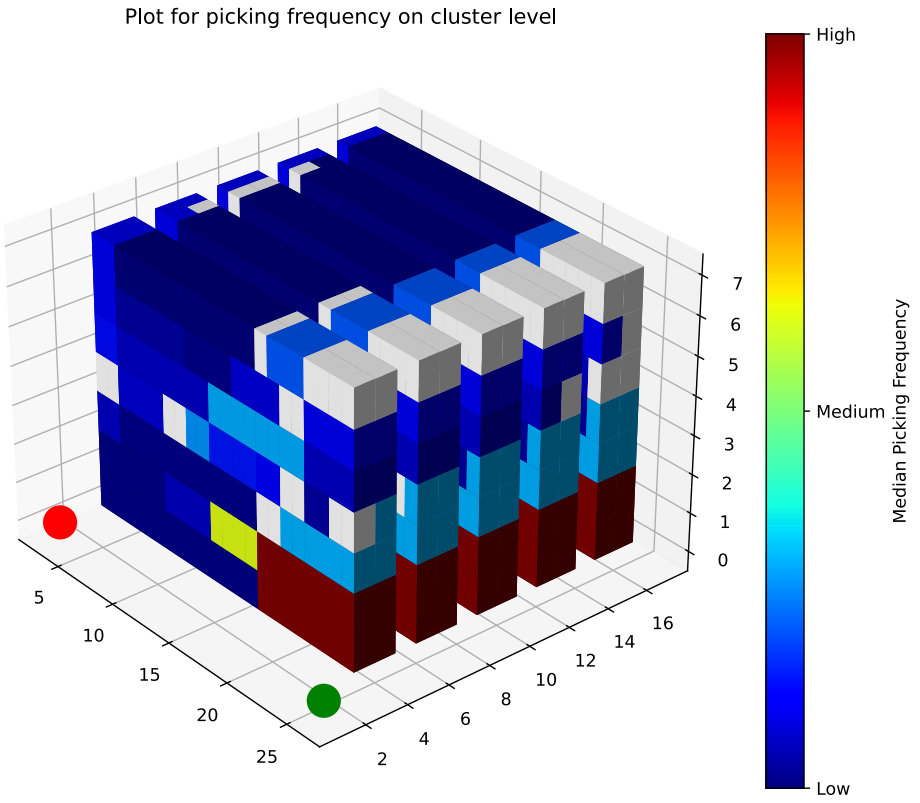

Figure 5.10: Median picking frequency for the case of optimizing the clusters for the best picking distance (on cluster- and product-level).

5.2.2 Three Clusters Using K-Medoids

The second scenario involves using 3 clusters created with K-Medoids clustering, similar to the 25-cluster case. Initially, the solution based on cluster-level optimization (see Figure 5.11) reveals a challenge when optimizing with fewer clusters: the solutions tend to perform similarly in terms of distance and ergonomics. This is due to the limited number of possible combinations in the program's logic, which is only 6 ($3 \cdot 2 \cdot 1$). Additionally, clusters with varying frequencies can impact performance, with high-frequency clusters at the bottom leading to better ergonomic and distance scores when the clusters are larger.

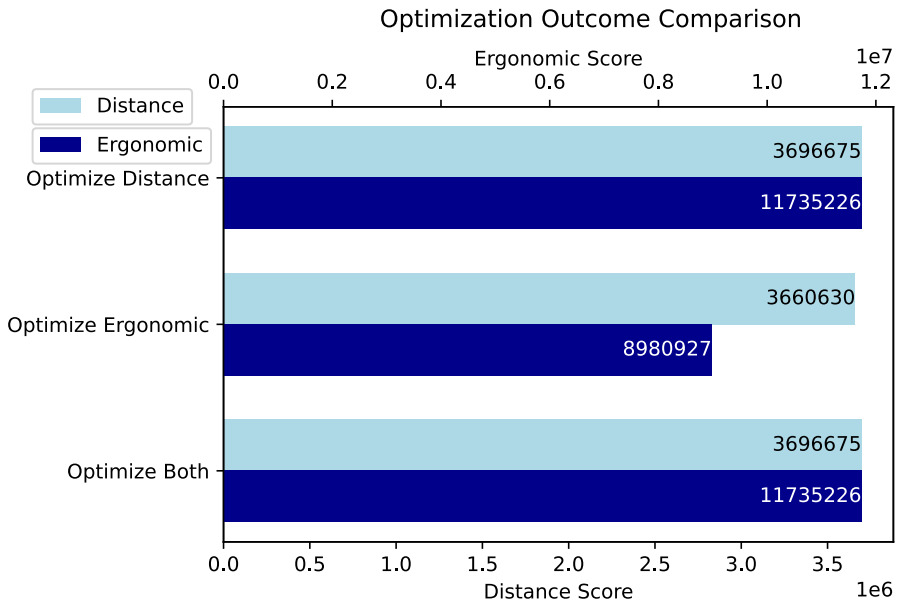

Figure 5.11: Plot of original optimization values when using few clusters.

Similar parameters were used for this optimization as for the former case presented, the difference is the number of clusters (see Table 5.3).

Table 5.3: Input variables for optimizing using 3 clusters and K-Medoids clustering.

Number of Clusters	3
Population Size	20
Number of Generations	10
Distance Share	0.9
Ergonomics Share	0.1

A greater volatility can be seen in the relative improvements when optimizing the solutions within the clusters (see Table 5.4) compared to the case of 25 clusters. For this case, the best ergonomic score was actually created when optimizing the combined solution on cluster-level for the best ergonomics, with an ergonomic score reduction of 42.6%, which was the highest relative reduction in ergonomic score for all of the three clustering cases. One thing to notice is that when the ergonomic score improved, the distance score tended to worsen, a trend more pronounced in this clustering scenario than in the 25-cluster case.

Table 5.4: Best scores after optimizing product order compared to initial cluster positioning for each optimization goal when optimizing using 3 clusters created with K-Medoids.

Optimization Goal (Cluster-Level)	Optimization Goal (Product-Level)	Distance Score	Diff. vs Cluster-Level (Distance)	Ergonomic Score	Diff. vs Cluster-Level (Ergonomic)
Best Distance	*Best Distance*	3,481,842	-5.8%	9,359,974	-20.2%
Best Ergonomics	*Best Combined*	3,588,265	-2.0%	8,705,476	-3.1%
Best Combined	*Best Ergonomics*	3,722,740	+0.7%	6,740,946	-42.6%

Also, for this optimization, the median picking frequency was plotted for the case of optimizing for the best picking distance (see Figure 5.12). Once again, the solution seem feasible since the most frequently picked clusters are placed closer to the starting point of the picking rounds, in this case closer to the ground-level.

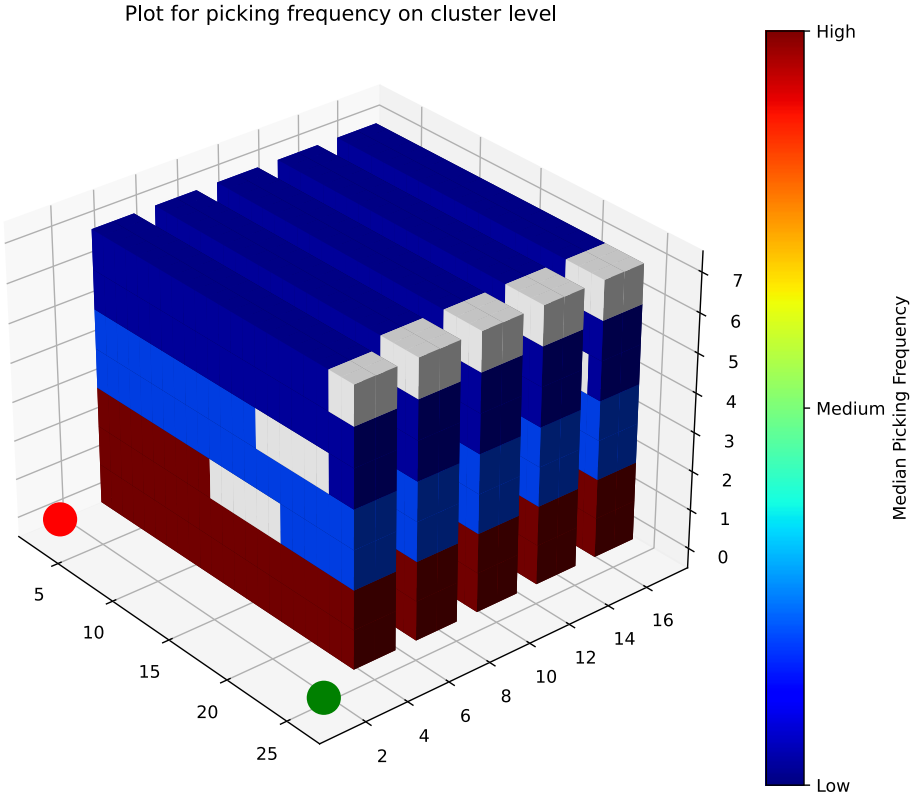

Figure 5.12: Median picking frequency for the case of optimizing the cluster positioning for the best picking distance (on cluster-level) using few clusters.

5.2.3 ABC-Classification

Finally, the clustering scenario when using the ABC-analysis is presented. Similar to K-Medoids creating 3 clusters, the ABC-analysis creates 3 clusters as mentioned earlier in the report. As with the 3 clusters created with K-Medoids, the initial solutions on cluster-level are fairly similar between the goal functions, with the optimization for distance and the combined goal function having the exact same outcome. This should be deemed reasonable since there are fewer optimization combinations available compared to the case with 25 clusters.

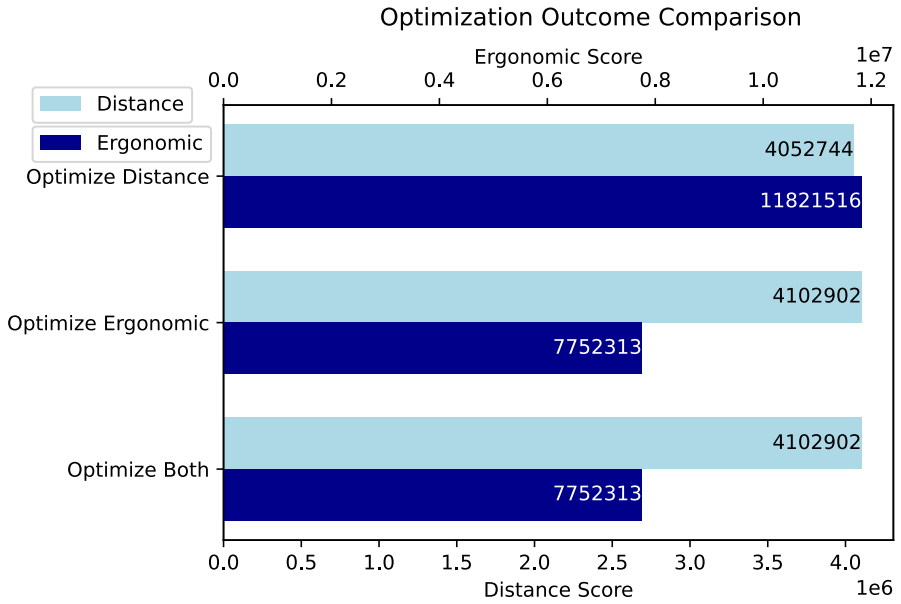

Figure 5.13: Plot of original optimization values when using ABC clusters.

Variables used for optimizing the ABC-clusters are the same as for the 3 clusters created with kmedoids (see Table 5.5).

Table 5.5: Input variables for optimizing using ABC-clustering.

Number of Clusters	*ABC*
Population Size	20
Number of Generations	10
Distance Share	0.9
Ergonomics Share	0.1

Optimizing the ABC-clusters on product-level yielded the most significant relative improvement in performance compared to the two other cases, considering that all outcomes of internal order optimization generated a better solution than the initial one for both variables (see Table 5.6). One thing to notice is the overall high, and consistent reduction in distance score (compared to the other clustering cases). This while still also improving the ergonomic score to a similar or higher degree than the other two cases.

Table 5.6: Best scores after optimizing product order compared to initial cluster positioning for each optimization goal when optimizing using ABC-clustering.

Optimization Goal (Cluster-Level)	Optimization Goal (Product-Level)	Distance Score	Diff. vs Cluster-Level (Distance)	Ergonomic Score	Diff. vs Cluster-Level (Ergonomic)
Best Distance	*Best Distance*	3,215,474	-20.7%	10,540,813	-10.8%
Best Ergonomics	*Best Ergonomics*	3,606,698	-12.1%	5,241,808	-32.4%
Best Combined	*Best Ergonomics*	3,428,052	-16.4%	5,401,904	-30.3%

The ABC-classification is also reasonable since the clusters with higher picking frequency are closer to the starting point than the lesser frequently picked clusters (see Figure 5.14). For this case, closer to the starting point means closer to the floor of the warehouse.

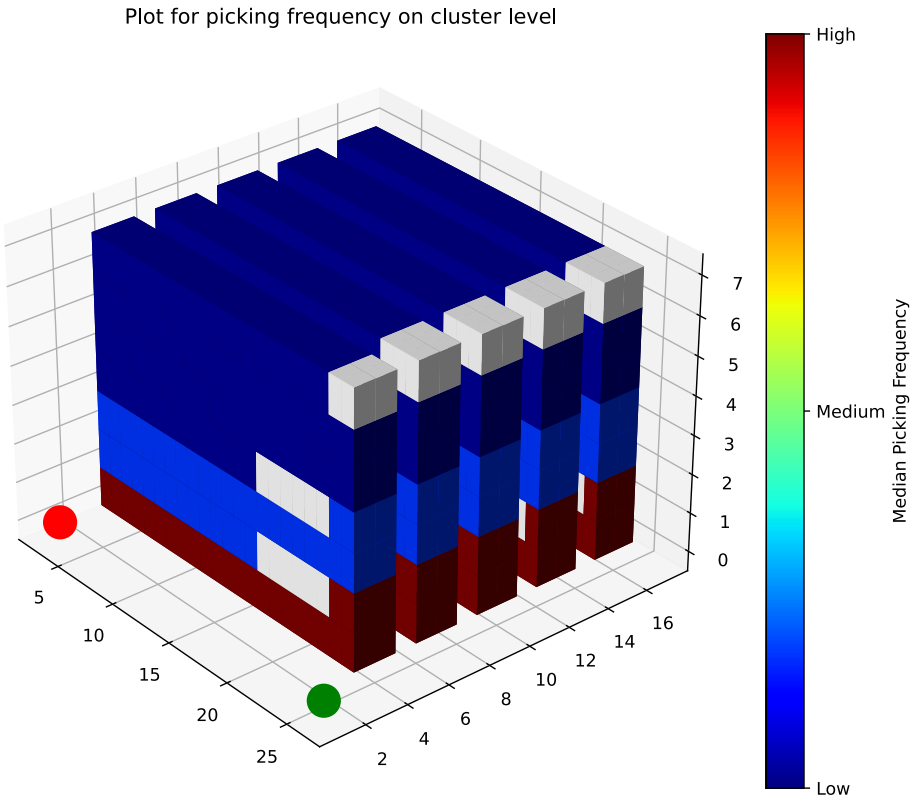

Figure 5.14: Median picking frequency for the case of optimizing the cluster positioning for the best picking distance (on cluster-level) using ABC clusters.

5.2.4 Comparison Between the Three Clustering Cases

In the final part of results, the three clustering cases are compared with respect to the best outcome for the different optimization goals (distance, ergonomics, and

combined). For each optimization variable, the best case when optimizing on the product-level is being presented for the different clustering-cases.

The initial comparison is for the case of optimizing for distance (see Figure 5.15). It can be seen that the best distance is found when having 25 clusters, and that the difference in distance score between the cases with 3 clusters is fairly small.

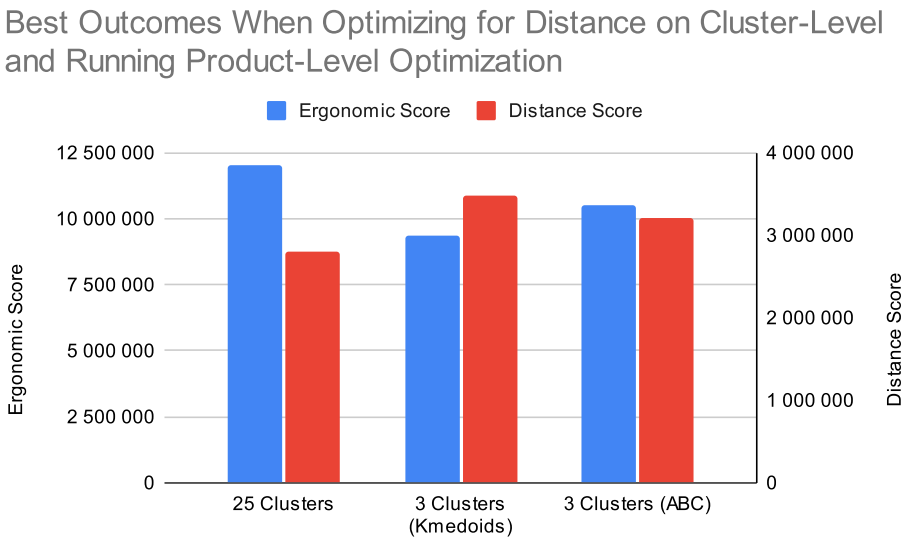

Figure 5.15: Comparison of the lowest distance-score for the three clustering cases when optimizing for distance on cluster-level and running optimization on product-level as well.

Compared to the random storage solution, all cases show better performance (see Figure 5.16). The random case was generated by using the same product placement logic as for the genetic algorithm but inserting a random product order and cluster order. It was repeated 10 times, and the average scores were calculated (13,289,293 for the ergonomic score and 3,810,064 for the distance score).

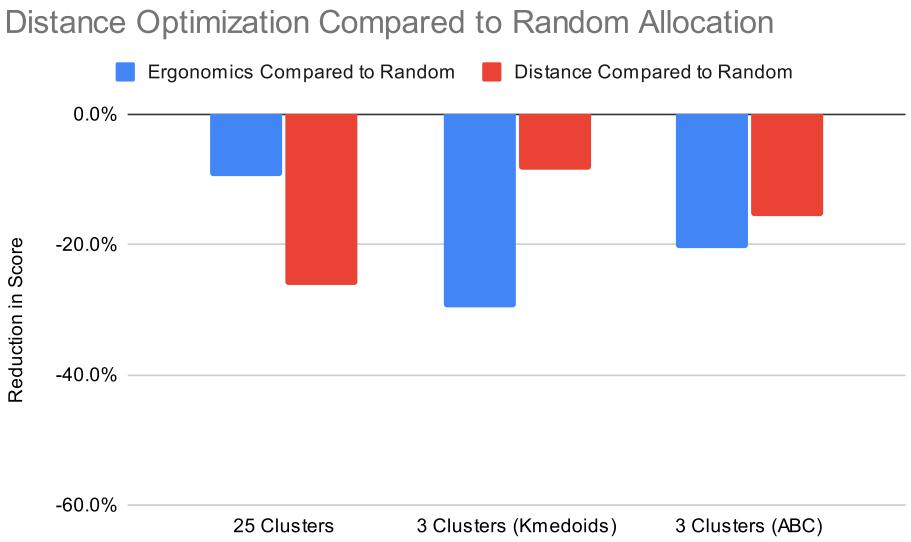

Figure 5.16: Comparison of the best lowest distance score results compared to the random storage allocation.

The second comparison considers the ergonomics as the variable optimized for (see Figure 5.17). While the case with 25 clusters performs better with respect to distance, the ABC-clustering manage to get the best outcome with respect to ergonomics.

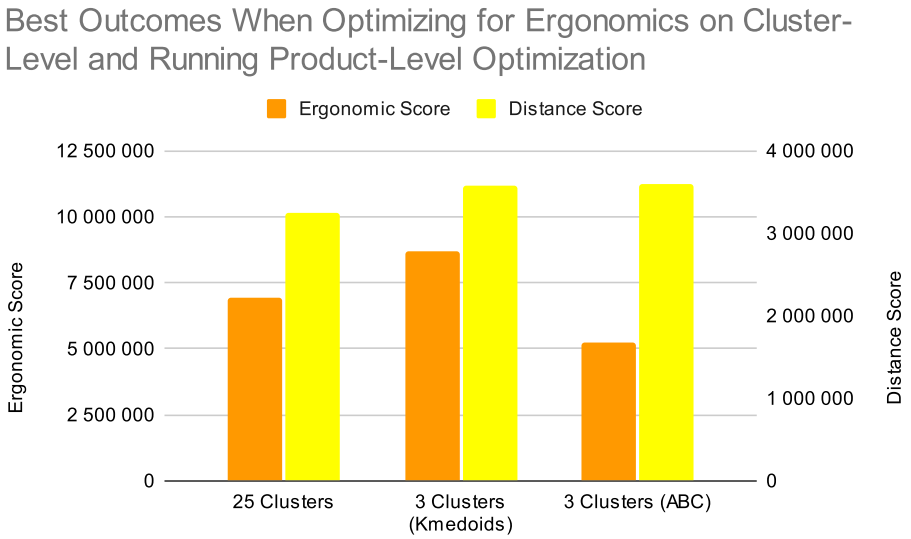

Figure 5.17: Comparison of the lowest ergonomics-score for the three clustering cases when optimizing for ergonomics on cluster-level and running optimization on product-level as well.

Solutions for the ergonomics are compared to the case of random storage allocation (see Figure 5.18). The plot shows that all these solutions perform better than the random allocation.

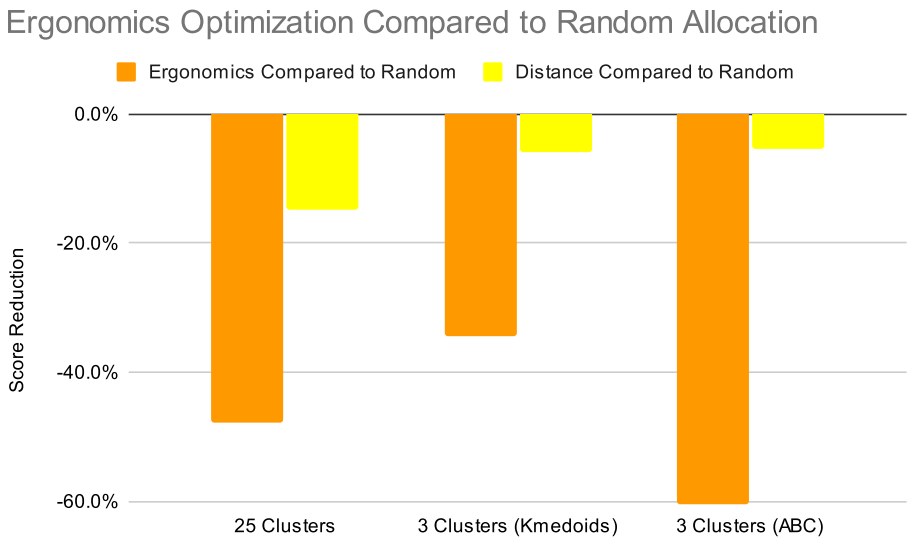

Figure 5.18: Comparison of the best lowest distance score results compared to the random storage allocation.

The third and final comparison is for the combined case. Regarding ergonomics, the cases with three clusters outperform the 25 cluster-case, but with respect to distance, the 25 cluster-case performs better than the other two.

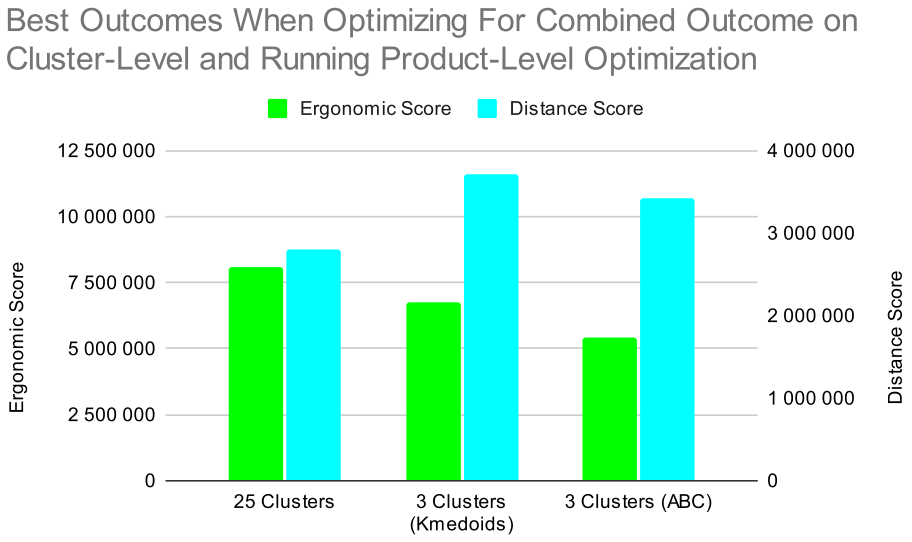

Figure 5.19: Comparison of the lowest combined-score for the three clustering cases when optimizing for the combined score on cluster-level and running optimization on product-level as well.

For the final solution, all cases perform better than the random allocation of products (see Figure 5.20).

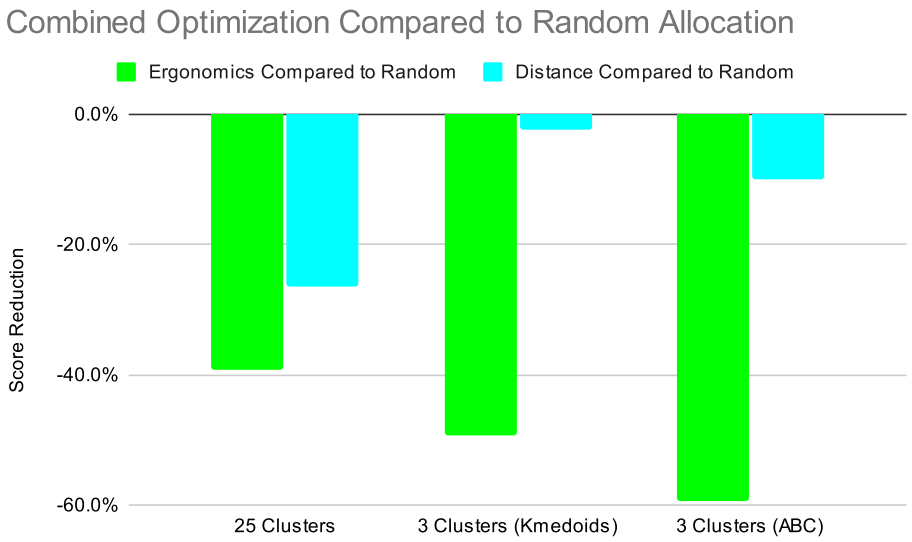

Figure 5.20: Comparison of the best lowest distance score results compared to the random storage allocation.

In summary, when using 25 clusters the outcomes tend show less variability when comparing the optimization results on cluster-level versus on the product-level. One reason for this could be the lower number of possible combinations when placing out the clusters in the first step of the optimization compared to the case of having 25 clusters, reducing ability to adapt to the goal function. Additionally, all presented optimizations on product-level perform significantly better than the random allocation of items.

6
Analysis

This chapter builds on the implementation and results presented earlier, integrating them with the theoretical foundation laid out in the report. It is organized into three sections: an analysis of the logic of product placement, an examination of clustering, and the final section focusing on the optimization.

6.1 Logic of Placing Out Products

The first area of analysis focuses on the logic behind the placement of clusters in the warehouse. This aspect is crucial for overall performance, as the placement of clusters affects their proximity to the starting point of picking rounds and their ergonomic efficiency. As mentioned earlier, clusters can be ”standing” or ”laying.” Using ”standing” clusters allows for more precise optimization of cluster positioning for picking distance. However, then the ergonomics would have been more random in its nature since the order of the products within the clusters (without internal optimization) are only based on product number (before internal optimization).

Previous literature emphasizes the importance of practical logic in real-world applications (Li et al., 2016; Yang & Nguyen, 2016). The clustering and optimization algorithms used are typically standard variants, but their effectiveness relies on the input from warehouse-specific logic, which must be tailored to function correctly. This study highlights the need to balance practical logic with complexity, ensuring the product placement logic is as realistic as possible without unnecessary complications, such as an overly advanced system for cluster placement.

Another consideration is the picking routes used to determine picking distance. This study employs a simple approach to calculating picking rounds. While more sophisticated methods, like the S-route, could better reflect real-world scenarios, the simpler approach is justified as the focus is on the storage location assignment problem, not route planning. Route planning is a distinct area of research within warehouse operations, and incorporating the testing of different routes into the slotting optimization would require additional time and resources.

6.2 Clustering

The visual inspection of the clustering results, in Figures 5.5, 5.6, and 5.7, reveals notable differences in cluster separation and overlap across the different scenarios. The overall characteristics of the clusters are determined based on the chosen features, with some clusters being for example characterized by high frequency and weight, while others exhibit the opposite traits. When the number of clusters is few, the characteristics become clearer and more distinct. However, as the number of clusters increases, it becomes increasingly difficult to separate the different clusters based on their characteristics.

A clear difference is observed when comparing the results obtained from using K-Mediods and ABC-analysis. While K-Mediods produced separated clusters with distinct boundaries, the ABC-analysis resulted in clusters with a significant overlap.

A closer examination of the clustering outcome with three clusters using K-Medoids, as illustrated in Figure 5.6, reveals interesting insights. Cluster 1 serves as a middle ground between cluster 2 & 3 in terms of features used for clustering. Cluster 2 & 3 have more extreme values with cluster 3 having higher picking frequency than the two others as an example. All three clusters are similar with respect to the number of products assigned to them.

The clustering results obtained from ABC-analysis, as shown in Figure 5.7, display significant differences compared to the outcomes generated by the K-Medoids algorithm with three clusters. Two significant differences are apparent: first, a greater overlap exists between the clusters, suggesting reduced separation; second, there is a more pronounced difference in cluster sizes compared to the K-Medoids clustering approach. This is since the ABC-clustering only considers picking frequency as the variable for clustering, resulting in the two other parameters not being taken into consideration.

Moreover, comparing Figures 5.5 and 5.6, it becomes apparent that there was no noticeable difference observed when varying the number of clusters. This observation is further supported by examining the plots of the silhouette score, Davies-Bouldin index, and the Calinski-Harabasz index (Figures 5.2, 5.3, and 5.4), as these scores remain relatively consistent across different cluster numbers.

It is crucial to explore various scenarios while aligning the clustering process with the intended goal (Hastie et al., 2009). As mentioned by Hastie et al. (2009), the two main goals are data segmentation, aimed at partitioning a database based on specific criteria, or providing descriptive statistics, focused on identifying natural groupings within the data where the number of groups is unknown and requires estimation.

For this study, the primary goal of clustering was data segmentation rather than determining the optimal number of clusters. The focus was on effectively group-

ing products based on their similarities rather than solely optimizing the cluster count. Thus, emphasis was placed on ensuring interpretability, ensuring that products within each cluster shared similar properties. In this context, feature selection played a crucial role, as selecting the right variables was essential for meaningful clustering outcomes. A significant aspect of clustering is choosing variables, only variables with a strong justification for their ability to distinctly define the clusters should be included in the analysis. (Columbia University Mailman School of Public Health, n.d.; Everitt et al., 2011; Hastie et al., 2009).

An important observation from the analysis of the result is that increasing the number of clusters leads to improved optimization for the genetic algorithm (on cluster-level). This improvement stems from the ability to move more clusters within the algorithm. While this trade-off may not be significant when solely using the clustering module, it becomes crucial when integrating both modules. The impact of this trade-off is evident in Figure 5.15, Figure 5.17, and Figure 5.19, where using 25 clusters yields better distance scores across all three cases, compared to using three clusters. Additionally, the figures illustrate that increasing the number of clusters enhances the stability of the solution, comparing to using only three clusters.

In summary, these findings highlight the importance of selecting an appropriate clustering approach. The key is to consider the overarching goal for clustering and tailor the approach. This ensures that the chosen method aligns effectively with the specific goals of the analysis, resulting in more purposeful insights.

6.2.1 Investigation of the Dataset

During the investigation of the dataset, it became evident that there was only a small correlation between products and orders. Consequently, it was decided not to factor in any correlation between products and orders when clustering the products due to the limited support for the orders and the small number of frequently ordered products. Potentially, this outcome could have been different if there were fewer products, for example in a different industry.

Given that clustering was performed in isolation, an order analysis could be included before the clustering algorithm to choose variables for clustering more efficiently. This modular approach of the program allows for future development if other use-cases present a more significant correlation between products and orders. Previous studies have shown that using order correlation can improve warehouse efficiency (R.-Q. Zhang et al., 2019), making it a logical addition when supported by the data. However, if the data does not support this method, its implementation is unlikely to be beneficial since only a small number of products would be considered in such an analysis. Furthermore, incorporating order correlation with low support might introduce null variables into the clustering analysis, negatively impacting the creation of distinct clusters.

6.3 Optimization

When analysing the results from the final algorithm, which combines the clustering and the genetic algorithm, there are several key takeaways. The biggest one is the consistent improvement observed when comparing random assignment with the utilization of a genetic algorithm across all cases.

Another takeaway is that double optimization may not be optimal for all cases, as seen in Table 5.2. The table shows that there are two cases when double optimization resulted in worse outcomes. Even if it is only these two cases in which it appeared, it is worth considering when using the algorithm. Therefore, comparing single optimizing with double optimization is key to get a deeper understanding of how to handle the data and strategy selection. This could also be a case of how times the genetic algorithm is allowed to be run with respect to population size and number of generations. In cases with more possible solutions (such as many clusters or optimizing on product-level), then these two parameters impact the outcome, since the lower the parameters are, the less likely it is for the algorithm to find an optimal solution.

Additionally, a notable observation from the results is the similarity between outcomes obtained through ABC-analysis and clustering with K-Medoids. It suggests that the specific grouping of products may be less critical than the arrangement of these groups for achieving optimal results.

Another consideration is if it is class-based storage when determining internal arrangement of products within a cluster. One could argue that the solution has converged to a more dedicated approach. As the initial approach was to not converge to a dedicated solution, as it may restrict flexibility and dynamic abilities (Li et al., 2016; Waubert de Puiseau et al., 2022) there are several benefits with also considering product order within clusters. For the scenario with fewer clusters, optimizing at the product level resulted in better scores, see Table 5.2. Another reason is that without investigating a dedicated approach it can sometimes be challenging to assume reliable results for optimizing only the cluster order.

Lastly, a notable challenge lies in balancing the optimization of distance and ergonomics. Often, optimization for one variable comes at the expense of the other, as illustrated in Figure 5.15, Figure 5.17, and Figure 5.19. Therefore, finding the right balance between these objectives can be seen as an ongoing task and will differ from case to case. Since this reason is rooted in the size of the different scores, the weights for each part in the combined case will differ greatly between cases. For example, if a warehouse is big i size, the distance score will become high. In contrast, if the inputs for the ergonomics is given high values, then the ergonomic score becomes high. These two examples highlight the alternating nature of the combined score.

6.4 Revisiting the Research Questions

To conclude the analysis, the research questions posed at the beginning of the thesis are revisited. As presented in the beginning of the thesis, the aim was to explore the intersection of artificial intelligence (AI) and warehouse management, focusing on two primary research questions.

RQ 1 How can AI-driven solutions optimize the storage location assignment problem (SLAP) to reduce travel distance, while optimizing the ergonomics for workers?

Following an analysis of the implemented algorithms and the outcomes, several insights have been identified. Firstly, the integration of AI, particularly genetic algorithms, has demonstrated significant improvements in optimizing the storage location assignment problem. The utilization of genetic algorithms consistently outperformed random assignment, showcasing the potential of AI-driven solutions to enhance efficiency in warehousing. Furthermore, clustering using the K-Medoids algorithm, as opposed to the ABC analysis, produced more well-defined and evenly distributed clusters. This further showcases the potential of AI-driven solutions in improving warehouse management, by creating more efficient and balanced groupings of products. The ABC analysis used for optimization yielded similar outcomes to as when the K-Medoids clustering was deployed (see Table 5.6 & 5.4).

RQ 2 What are the challenges and considerations in implementing AI-supported slotting optimization solutions in warehouses?

The analysis highlighted the importance of carefully considering optimization strategies, particularly in instances of double optimization (at both cluster and product levels). While double optimization may yield favorable results in some cases, it is not universally applicable. A comparison between single and double optimization is necessary to determine the most suitable approach for specific scenarios. For instance, if data is difficult to cluster into many distinct clusters, fewer clusters could be used and optimizing the solution on product-level. This approach should be seen more as a dedicated storage solution, in contrast to the optimization on cluster-level which can be seen as class-based storage.

Moreover, the similarity between outcomes obtained through ABC-analysis and clustering with K-Medoids underscores the importance of grouping products effectively rather than solely focusing on the clustering method. This suggests that the arrangement of product groups plays a crucial role in achieving optimal results. Additionally, it highlights that when it comes to clustering, traditional methods, such as ABC-analysis, can generate feasible solutions while being easier than AI-clustering to implement.

Another aspect to consider is the use of optimizing on product-level, which can become rigid. While dedicated storage offers advantages in terms of optimization, such

as improved scores in scenarios with fewer clusters, it may also restrict flexibility and dynamic abilities due to demand of product changing after the optimization being made. Therefore, striking a balance between dedicated clustering approaches and maintaining flexibility is essential for optimizing placements effectively. The user of the program must know the objectives of the situation, otherwise a more dedicated storage solution could hinder a more long-term optimization, which could be solved by focusing on the class-based approach, optimizing only on cluster-level.

Finally, a significant challenge lies in balancing the optimization of distance and ergonomics. The trade-off between these objectives needs careful consideration and ongoing adjustments to find the optimal balance for each unique warehouse environment.

6.5 Future Research

Areas for future research could be to improve the flexibility of cluster positioning, creating a more precise distance calculation, and look further into finding a balance when optimizing for distance and ergonomics.

Improving flexibility for cluster positioning should be considered since that could potentially enhance the performance of the program when having fewer clusters. The presented logic is fairly rigid when it comes to placing out the clusters, especially when using a lower amount of clusters, such as three.

Furthermore, investigating the optimal number of clusters could also be an area of focus for future research. To truly enhance the efficiency of the clustering algorithm, it is crucial to find a way to incorporate cluster number in to the equation. This aspect is essential as it directly impacts the performance and accuracy of the algorithm, ensuring that it can effectively group data into meaningful clusters.

A more precise distance calculation could enhance the simulation with respect to how the picking rounds are made in reality. The simple approach taken in this report for distance calculation, not using something more like reality, such as the return method or S-shape hinders a more real-world like simulation.

The balance for the solution considering distance and ergonomics is also worth looking further into. This could involve automatically finding a trade-off solution without manual input that requires trial-and-error to find a wanted balance, reducing the need for more in-depth knowledge about operating the program.

7
Conclusion

In conclusion, the combined use of clustering and a genetic algorithm effectively addressed the Storage Location Assignment Problem (SLAP) by reducing travel distance and enhancing worker ergonomics. Compared to a random storage assignment, the proposed solution reduced the picking distance score by up to 25% and the ergonomic score by up to 60%.

When implementing the AI-solutions, it is key to understand the objectives of the situation. Without clear objectives, the solution can become rigid, creating more of a dedicated storage solution. Also, when developing such a program, understanding of warehouse operations is key for success.

This research provides insights to the application of AI-driven solutions in warehouse management, offering guidance for optimizing operational efficiency considering factors for travel distance, ergonomics, and implementation challenges. Future research could include improving the distance calculation to better reflect a real-world warehouse. Through continued research and refinement, AI-supported slotting optimization solutions have the potential to further enhance warehousing practices and drive improvements in efficiency, and health of workers.

References

Amazon Web Services. (n.d.). Easily add intelligence to your applications. Retrieved May 28, 2024, from https://aws.amazon.com/machine-learning/ai-services/

Anyoha, R. (2017, August 28). *The History of Artificial Intelligence*. Retrieved May 28, 2024, from https://sitn.hms.harvard.edu/flash/2017/history-artificial-intelligence/

Bell, E., Bryman, A., & Harley, B. (2019). *Business research methods* (5th ed.). Oxford University Press.

Blum, C., & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Comput. Surv.*, *35*(3), 268–308. https://doi.org/10.1145/937503.937505

Britannica. (2023, December 29). *Work*. Retrieved May 28, 2024, from https://www.britannica.com/science/work-physics

Bustamante, F., Dekhne, A., Herrmann, J., & Singh, V. (2020, February 6). *Improving warehouse operations—digitally*. Retrieved May 28, 2024, from https://www.mckinsey.com/capabilities/operations/our-insights/improving-warehouse-operations-digitally

Calzavara, M., Glock, C. H., Grosse, E. H., Persona, A., & Sgarbossa, F. (2017). Analysis of economic and ergonomic performance measures of different rack layouts in an order picking warehouse. *Computers & Industrial Engineering*, *111*, 527–536. https://doi.org/10.1016/j.cie.2016.07.001

Cambridge Dictionary. (n.d.). *Warehousing*. Retrieved May 28, 2024, from https://dictionary.cambridge.org/dictionary/english/warehousing

Columbia University Mailman School of Public Health. (n.d.). *K-Means Cluster Analysis*. Retrieved May 28, 2024, from https://www.publichealth.columbia.edu/research/population-health-methods/k-means-cluster-analysis

Davidson, B., & Patel, R. (2019). *Forskningsmetodikens grunder*. Studentlitteratur.

de Koster, R., Le-Duc, T., & Roodbergen, K. J. (2007). Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, *182*(2), 481–501. https://doi.org/10.1016/j.ejor.2006.07.009

Ermel, A. P. C., Lacerda, D. P., Morandi, M. I. W. M., & Gauss, L. (2021). *Literature Reviews: Modern Methods For Investigating Scientific And Technological Knowledge* (1st ed.). Springer.

Everitt, B. S., Landau, S., Leese, M., & Stahl, D. (2011). *Cluster analysis* (5th ed.). Wiley.

Gu, J., Goetschalckx, M., & McGinnis, L. F. (2007). Research on warehouse operation: A comprehensive review. *European Journal of Operational Research*, *177*(1), 1–21. https://doi.org/10.1016/j.ejor.2006.02.025

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (2nd ed.). Springer.

Jiang, W., Liu, J., Dong, Y., & 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. https://doi.org/10.1080/00207543.2020.1766714

Jo, T. (2021). *Machine learning foundations: Supervised, unsupervised, and advanced learning*. Springer Nature Switzerland AG.

Jonsson, P. (2008). *Logistics and supply chain management*. McGraw-Hill Education.

Lesch, V., Müller, P. B., Krämer, M., Hadry, M., Kounev, S., & Krupitzer, C. (2023). Optimizing storage assignment, order picking, and their interaction in mezzanine warehouses. *Applied Intelligence*, 1–25. https://doi.org/10.1007/s10489-022-04443-x

Li, J., Moghaddam, M., & Nof, S. (2016). Dynamic storage assignment with product affinity and ABC classification - A case study. *The International Journal of Advanced Manufacturing Technology*, *84*. https://doi.org/10.1007/s00170-015-7806-7

Lorenc, A., Kuźnar, M., & Lerher, T. (2021). Solving product allocation problem (PAP) by using ANN and clustering. *FME Transactions*, *49*(1), 206–213. https://doi.org/10.5937/fme2101206L

Lorenc, A., & Lerher, T. (2020). Pickupsimulo–Prototype of Intelligent Software to Support Warehouse Managers Decisions for Product Allocation Problem. *Applied Sciences*, *10*(23). https://doi.org/10.3390/app10238683

Marr, B. (2023, May 19). *A Short History Of ChatGPT: How We Got To Where We Are Today*. Retrieved May 28, 2024, from https://www.forbes.com/sites/bernardmarr/2023/05/19/a-short-history-of-chatgpt-how-we-got-to-where-we-are-today/?sh=57a9e240674f

Marras, W. S., Granata, K. P., Davis, K. G., Allread, W. G., & Jorgensen, M. J. (1999). Effects of box features on spine loading during warehouse order selecting. *Ergonomics*, *42*(7), 980–996. https://doi.org/10.1080/001401399185252

Medrano-Zarazúa, L., Saucedo-Martínez, J. A., & Bolaños-Zuñiga, J. (2024). Storage Location Assignment Problem in a Warehouse: A Literature Review. In J. A. Marmolejo-Saucedo, R. Rodríguez-Aguilar, P. Vasant, I. Litvinchev, & B. M. Retana-Blanco (Eds.), *Computer science and engineering in health services* (pp. 15–37). Springer International Publishing.

Nafar, M. R. (2022). *A study on storage allocation problem based on clustering algorithms for the improvement of warehouse efficiency* [Master's thesis]. Concordia University [John Molson School of Business].

NIST. (n.d.). *What are outliers in the data?* Retrieved May 28, 2024, from https://www.itl.nist.gov/div898/handbook/prc/section1/prc16.htm

Pang, K.-W., & Chan, H.-L. (2017). Data mining-based algorithm for storage location assignment in a randomised warehouse. *International Journal of Production Research*, *55*(14), 4035–4052. https://doi.org/10.1080/00207543.2016.1244615

Placek, M. (2022, September 28). *Number of warehouses worldwide from 2018 to 2025, by region*. Retrieved May 28, 2024, from https://www-statista-com.eu1.proxy.openathens.net/statistics/1223289/warehouses-worldwide-region/

Pournader, M., Ghaderi, H., Hassanzadegan, A., & Fahimnia, B. (2021). Artificial intelligence applications in supply chain management. *International Journal of Production Economics*, *241*, 108250. https://doi.org/10.1016/j.ijpe.2021.108250

Pruzan, P. (2016). *Research methodology: The aims, practices and ethics of science*. Springer International Publishing.

Rachwał, A., Popławska, E., Gorgol, I., Cieplak, T., Pliszczuk, D., Skowron, Ł., & Rymarczyk, T. (2023). Determining the quality of a dataset in clustering terms. *Appl. Sci.*, *13*(5), 2942. https://doi.org/10.3390/app13052942

Rainie, L., Funk, C., Anderson, M., & Tyson, A. (2022, March 17). *1. How Americans think about artificial intelligence*. Retrieved May 28, 2024, from https://www.pewresearch.org/internet/2022/03/17/how-americans-think-about-artificial-intelligence/

Raschka, S. (n.d.). *apriori: Frequent itemsets via the Apriori algorithm*. Retrieved May 28, 2024, from https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/apriori/

Reilly, A., Depa, J., & Douglass, G. (2019). *AI: BUILT TO SCALE - From experimental to exponential*. Retrieved May 28, 2024, from https://www.accenture.com/content/dam/accenture/final/a-com-migration/custom/_acnmedia/thought-leadership-assets/pdf-2/Accenture-Built-to-Scale-PDF-Report.pdf#zoom=50

Rudden, J. (2024, January 9). *Leading risks to large, mid-size and small companies globally for 2023*. Retrieved May 28, 2024, from https://www-statista-com.eu1.proxy.openathens.net/statistics/422207/leading-business-risks-by-company-size/

Rushton, A., Croucher, P., & Baker, P. (2021). *The handbook of logistics and distribution management: Understanding the supply chain*. Kogan Page.

Russell, S., & Norvig, P. (2009). *Artificial Intelligence: A Modern Approach* (3rd ed.). Prentice Hall Press.

Schuhmacher, J., & Hummel, V. (2023). Development of an AI-Based Method for Dynamic Affinity-Based Warehouse Slotting Using Indoor Localisation Data. In K. von Leipzig, N. Sacks, & M. Mc Clelland (Eds.), *Smart, sustainable manufacturing in an ever-changing world* (pp. 149–160). Springer International Publishing.

scikit-learn. (n.d.-a). *sklearn.cluster.AgglomerativeClustering*. Retrieved May 28, 2024, from https://scikit-learn.org/stable/modules/generated/sklearn.cluster.AgglomerativeClustering.html

scikit-learn. (n.d.-b). *sklearn.cluster.KMeans*. Retrieved May 28, 2024, from https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

scikit-learn. (n.d.-c). *sklearn.preprocessing.QuantileTransformer*. Retrieved May 28, 2024, from https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.QuantileTransformer.html

scikit-learn-extra. (n.d.). *sklearn_extra.cluster.KMedoids*. Retrieved May 28, 2024, from https://scikit-learn-extra.readthedocs.io/en/stable/generated/sklearn_extra.cluster.KMedoids.html

Silva, A., Roodbergen, K. J., Coelho, L. C., & Darvish, M. (2022). Estimating optimal ABC zone sizes in manual warehouses. *International Journal of Production Economics*, *252*, 108579. https://doi.org/10.1016/j.ijpe.2022.108579

Smith, P. (2022, May 3). *Leading supply chain management improvements brands are investing in worldwide in 2021 and 2022*. Retrieved May 28, 2024, from https://www-statista-com.eu1.proxy.openathens.net/statistics/1305964/main-supply-chain-management-improvements-worldwide/

Spataro, J. (2023, March 16). *Introducing Microsoft 365 Copilot – your copilot for work*. Retrieved May 28, 2024, from https://blogs.microsoft.com/blog/2023/03/16/introducing-microsoft-365-copilot-your-copilot-for-work/

Swisslog. (n.d.). *The learning warehouse – the next quantum leap thanks to artificial intelligence*. Retrieved May 28, 2024, from https://www.swisslog.com/en-us/about-swisslog/our-offering/industry-40/artificial-intelligence-ai-machine-learning-warehouse

Talbi, E.-G. (2002). A taxonomy of hybrid metaheuristics. *Journal of Heuristics*, *8*, 541–564. https://doi.org/10.1023/A:1016540724870

Tompkins, J. A., White, J. A., Bozer, Y. A., Frazelle, E. H., & Tanchoco, J. M. A. (2003). *Facilities Planning* (3rd ed.). John Wiley & Sons.

Veres, P. (2023). Increasing the efficiency of warehouse analysis using artificial intelligence. *Acta logistica*, *10*, 445–451. https://doi.org/10.22306/al.v10i3.415

Wahde, M. (2008). *Biologically inspired optimization methods: An introduction*. WIT Press.

Waters, T. R., Putz-Anderson, V., Garg, A., & Fine, L. J. (1993). Revised NIOSH equation for the design and evaluation of manual lifting tasks. *Ergonomics*, *36*(7), 749–776. https://doi.org/10.1080/00140139308967940

Waubert de Puiseau, C., Nanfack, D. T., Tercan, H., Löbbert-Plattfaut, J., & Meisen, T. (2022). Dynamic Storage Location Assignment in Warehouses Using Deep Reinforcement Learning. *Technologies*, *10*(6). https://doi.org/10.3390/technologies10060129

Xin, C., Liu, X., Deng, Y., & Lang, Q. (2019). An optimization algorithm based on text clustering for warehouse storage location allocation. *2019 1st International Conference on Industrial Artificial Intelligence (IAI)*, 1–6. https://doi.org/10.1109/ICIAI.2019.8850832

Yang, C.-L., & Nguyen, T. P. Q. (2016). Constrained clustering method for class-based storage location assignment in warehouse. *Industrial Management & Data Systems*, *116*(4), 667–689. https://doi.org/10.1108/IMDS-09-2015-0361

Zhang, R.-Q., Wang, M., & Pan, X. (2019). New model of the storage location assignment problem considering demand correlation pattern. *Computers & Industrial Engineering*, *129*, 210–219. https://doi.org/10.1016/j.cie.2019.01.027

Zhang, S., Fu, L., Chen, R., & Mei, Y. (2020). Optimizing the cargo location assignment of retail e-commerce based on an artificial fish swarm algorithm.

Mathematical Problems in Engineering, *2020*, 1–14. https://doi.org/10.1155/2020/5609873

Zhang, S., Fu, L., Wang, R., & Chen, R. (2020). The optimization of the location of the cargo in three-dimension shelf: Employing the FP-Tree and the Artificial Fish Swarm Algorithms. *Journal of Control Science and Engineering*, *2020*, 1–15. https://doi.org/10.1155/2020/8832691

A

Appendix - Table of Relevant Literature

Table A.1: Summary of selected literature within the field of warehousing slotting.

Article	Algorithm(s) Used	Research Approach	Identified Opportunities	Identified Challenges
(Schuhmacher & Hummel, 2023)	K-Means	1. Solution development. 2. Solution case testing.	Using AI for real-time data, and hybrid decision making in slotting.	Optimizing slotting with ergonomics in mind was not done.
(R.-Q. Zhang et al., 2019)	DCP, MIH	1. Solution development. 2. Solution testing on real case data.	In-depth demand correlation analysis used to optimize slotting.	The approach is only beneficial when having clear correlations.
(Xin et al., 2019)	Text clustering and correlation (BTC)	1. Algorithm development. 2. Algorithm testing on real case data.	Improved clustering and slotting performance when demand correlation is weak.	Do not work as effective when product correlation is strong.
(Jiang et al., 2021)	GA and PSO	1. Solution/algorithm development. 2. Test on generated dataset.	Using a hybrid algorithm to reduce computational time.	Not many slotting parameters were considered during optimization.
(Silva et al., 2022)	OLS, RT, RF, and MLP	1. Problem definition. 2. Algorithm testing on synthetic data.	Using AI-algorithms for automatic ABC-classification.	Only a few parameters were considered when training the models.
(Medrano-Zarazúa et al., 2024)	-	1. Literature review.	Insights to research within SLAP, solution methods, performance measures, and used constraints.	Creating a dynamic slotting policy to account for different settings.
(Li et al., 2016)	GA	1. Algorithm formulation. 2. Algorithm testing on real data.	Benefits in slotting from using product affinity and ABC classification.	Interdependencies change over time between products, resulting in poorer slotting performance.
(Yang & Nguyen, 2016)	Constrained clustering (semi supervised)	1. Analysis framework presentation. 2. Case study testing the framework	Using pre-defined variables for clusters considers more practical variables in slotting.	An even better solution could potentially be created with a hybrid algorithm.
(Veres, 2023)	ANN, black hole optimization	1. Model presentation. 2. Model implementation using case data.	Improved ABC-XYZ classification using AI based on manual classifications.	Can work differently between companies.
(Lesch et al., 2023)	GA, ACO	1. Model presentation. 2. Computational performance evaluation of model.	Incorporating economic and ergonomics factors in optimization slotting.	The model performs even better when warehouses are large.
(Waubert de Puiseau et al., 2022)	Deep Reinforcement Learning	1. Case presentation. 2. Algorithm presentation.	Algorithms performed better than ABC classification and random allocation. Does not require manual labels on the data.	Difficulty of handling new types of goods arriving.
(Lorenc et al., 2021)	ANN, clustering	1. Data analysis. 2. Model development and testing.	Reduction of around 10% in picking time when distances are greater.	No improvements in picking time when products are closer to each other.
(Lorenc & Lerher, 2020)	ANN	1. Model presentation. 2. Model implementation.	Using an application to foresee effects of taken actions enables decision support for managers when slotting items.	An advanced program is used for testing the algorithms.
(S. Zhang, Fu, Wang, & Chen, 2020)	FP-Tree, Artificial Fish Swarm Algorithm	1. Problem description. 2. Model creation. 3. Simulation using data.	Using a combined method for determining locations in a three-dimensional storage system where associations between products is a key factor.	A more optimal solution could be created if order picking is considered at the same time.
(S. Zhang, Fu, Chen, & Mei, 2020)	PSO, GA, Artificial Fish Swarm Algorithm	1. Problem description. 2. Model creation. 3. Numerical tests.	The proposed algorithm is more efficient than PSO and GA.	Only a few variables were considered for the slotting optimization.

DEPARTMENT OF TECHNOLOGY MANAGEMENT AND ECONOMICS
DIVISION OF SUPPLY AND OPERATIONS MANAGEMENT
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY