Optimal Order Batching for Automated Warehouse Picking

by

Zeynep Kucuksari

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Management Sciences

Waterloo, Ontario, Canada, 2023

© Zeynep Kucuksari 2023

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

With the unexpected increase in demand and the need to minimize human interaction during the Covid-19 pandemic, companies have been forced to accelerate the transition from traditional to robotic mobile fulfillment systems. The key to a successful warehouse management system, whether traditional or automated, is an efficient order-picking process. In this study, we focus on the order batching problem, where items and orders are grouped into batches for simultaneous picking in automated warehouses that use autonomous picking carts. We propose five different mathematical models, including a generalized quadratic assignment model. We focus on the latter as it provides the best results and propose a Lagrangian relaxation to obtain lower bounds and an iterative Simulated Annealing (SA) algorithm that generates an initial solution using a K-means clustering algorithm. We carry out testing using an open-source dataset to assess the iterative SA algorithm in minimizing congestion and travel distance in an automated warehouse. We find that it finds solutions of good quality as measured by Lagrangian relaxation and is capable of solving large realistic instances. The solutions successfully minimize travel distance and reduce congestion by limiting path intersections.

Acknowledgements

I would like to express my sincere gratitude to my thesis supervisor Prof. Samir Elhedhli for his constant support, patience, and understanding throughout the past two years. Without his encouragement and guidance, it would not be possible to complete this thesis.

In addition, I would like to thank Dr. Sibel Alumur Alev and Dr. Fatih Safa Erenay for their contributions and valuable comments as readers of this thesis.

I owe special thanks to my friends Esma Akgun and Cansu Dagsuyu for their encouragement, continuing support, and patience during my thesis writing stage. Their support kept me away from stress and helped me finish my thesis successfully.

Lastly, I am grateful to everyone who made this research possible.

Dedication

"There should be no boundaries to human endeavor. We are all different. However bad life may seem, there is always something you can do and succeed at. While there's life, there is hope."

-Stephen Hawking

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The transition from traditional warehouse systems to intelligent automated systems has been accelerated during the COVID-19 pandemic as companies faced a surge in online demand and a shortage in labor due to limited human interaction and mandatory distancing measures. For example, Amazon had already implemented an automated system using Kiva robotics that carries racks from storage areas to human-operated workstations to perform order picking before the pandemic (Salles, 2020). Amazon decided to replace most of its warehouse employees with robots in order to speed up deliveries and cope with increased demand. Automated warehouse systems enabled companies to collect data throughout every step of the order-handling process. The collected data carry valuable information that can be used to determine new strategies to improve efficiency and detect weaknesses in the system. This highlights the importance of automated warehouse systems for companies looking to survive in a post-pandemic world.

The key to a successful warehouse management system is having an efficient order fulfillment process that involves order picking, consolidation, and packing. Order picking is retrieving the listed items from their storage areas to the consolidation area in order to complete customer orders (Petersen and Schmenner, 2007). This process involves retrieving information on products to collect, assigning products to pick lists, collecting products from storage areas by human or robot pickers, receiving products in the consolidation area, and packing each order separately to make them ready for delivery. Order picking can be performed by humans or robots. As an example, we can look at online shopping to understand how order picking works in e-commerce companies. After receiving a customer's order, items are collected from the warehouse as quickly as possible for packing and delivery. In traditional warehouse systems, human workers use pick lists, handheld scanners, and other tools to find the location of items and collect them. In this system, a human picker starts from a dedicated workstation, collects items from the assigned list, and returns items to the consolidation area for packing. If the pick cycle is large, this system can mostly cause poor performance rate and delays (Boysen et al., 2018). Because of these potential limitations of traditional systems and recent developments in robotic solutions, warehouses had to consider and adopt automated systems. In contrast to traditional, automated warehouse systems need robots, scanners, charging ports, sensors, and other tools to perform efficient order picking. The efficiency and accuracy of order picking have a significant role in customer satisfaction and, therefore, a company's overall success.

Order picking is identified as the most expensive and challenging activity in a warehouse. Frazelle (2002) reported that the cost of order-picking activities represents more than 65% of the total warehouse operating cost. Inefficient order picking may result in higher costs from incorrect shipments and delayed deliveries. One of the biggest challenges is that order picking is expensive, has poor ergonomics, and needs a large number of operators who are well-trained and available (Azadeh et al., 2019). The Robotic Mobile Fulfilment System (RMFS) has provided online retailers a chance to improve order-picking efficiency and reduce labor expenses by enhancing order fulfillment and inventory control (Azadeh et al., 2019). In an RMFS, robots move products from racks or shelves to workstations where employees collect the needed items and place them in containers. The robots are equipped with sensors and cameras to assist them in navigating the warehouse and avoiding obstacles, and they can be programmed to collect certain products from predefined locations. With the help of RMFS technology, order fulfillment is becoming much faster and more effective since there is no need to spend more time searching the warehouse to find the listed items.

Several surveys have been published in recent years, reviewing the available technologies, research problems, solution methodologies, research gaps, and trends in automated warehouse systems. Autonomous mobile robots (AMRs) have gained significant attention, with Kiva robots being the most commonly used for “goods to person” AMRs. However, “Autonomous picking carts” have emerged as a new solution, which requires a different approach to order sequencing since these carts are not equipped with shelves. Rack movement remains a key issue for Kiva robots, but the sequencing of orders is a higher priority for autonomous picking carts. Figure 1.1 shows available autonomous picking carts in the industry.

Several policies, such as batch picking, are available to reduce the cost of order picking. Batch picking is grouping orders or items into batches for simultaneous collection. Items from the same batch are collected by the same robot, whereas some of the items in an order may have been placed by different batches and collected by different robots. However, if required, order integrity can be maintained in batch picking by assigning entire items in each order to the same batch. If order integrity is not satisfied, orders must be consolidated after collecting items from the storage area. Another policy is zone clustering, a storage assignment model combining items based on specific metrics such as frequency. This policy can help reduce the total traveled time of robots since frequently bought products will be close to workstations.

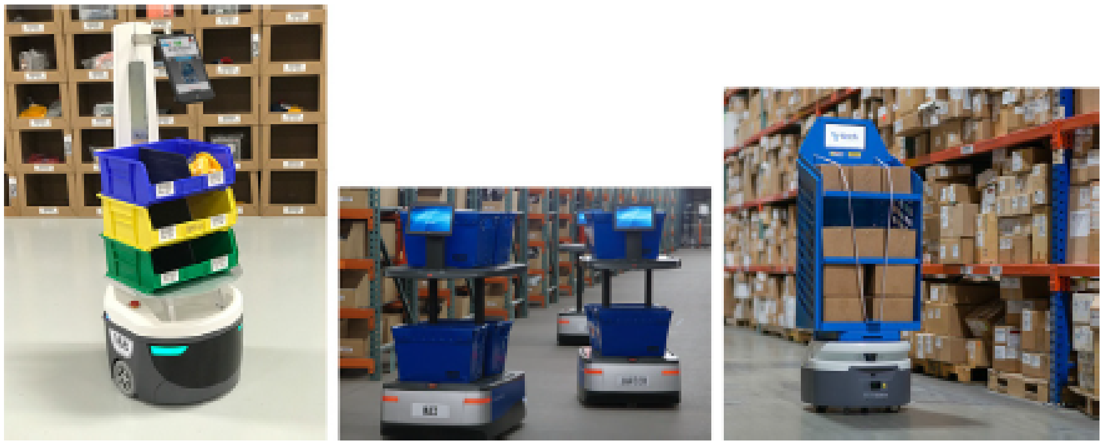

Figure 1.1: Example of autonomous picking cart from Locus Robotics, 6 River Systems, Fetch Robotics (Xie et al., 2022).

This study focuses on the order batching problem in an automated warehouse management system using autonomous picking carts. Order batching involves grouping multiple customer orders into batches to minimize travel time and maximize the efficiency of order picking. This problem is complex and challenging, requiring the optimization of various factors such as the number and size of batches, the sequence of orders within each batch, and the allocation of orders to autonomous picking carts. Several studies have proposed different algorithms and methodologies to tackle this problem, but it remains an active research area.

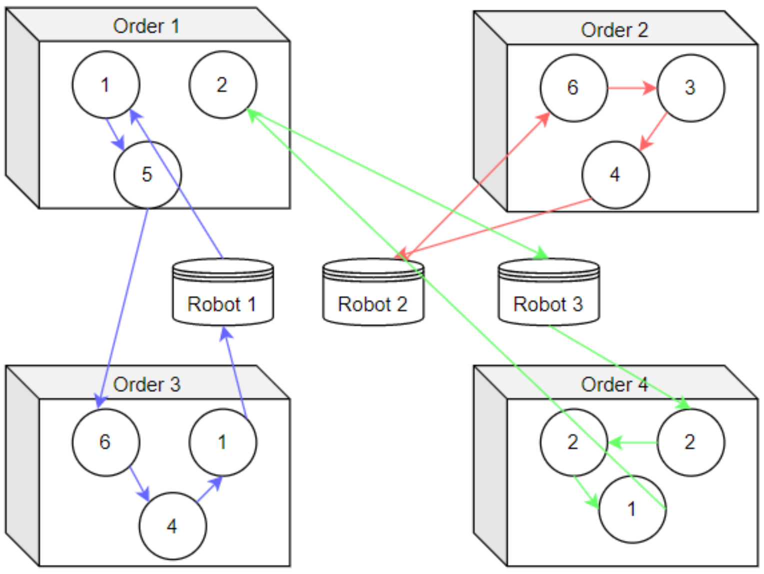

Figure 1.2: A small example of order batching and robot routing with four orders, six SKUs, and three robots.

The aim of this work is to optimize order batching by considering the unique characteristics and constraints of autonomous picking carts, such as limited capacity and the need for order sequencing. We focus on systems requiring human and robot collaboration, referred to as Meet-in-Isle, such as Locus Robotics. After assigning items to batches, each batch will be assigned to a robot that goes through the directional aisles of the warehouse to collect the items. When the robot arrives at the location of an item, the human picker responsible for the aisle will take the item from the rack, scan it, and place it in the robot's tray. Then, the robot will move on to the next item in the pick list. When all items are collected, the robot will return to the consolidation area to drop off items for packaging. After completing picking, robots will wait temporarily in the workstation area until they receive a new assignment.

Figure 1.2 demonstrates a small example of an order batching and routing system with four orders, six SKUs, and three robots. The goal is to assign items to a batch and each batch to a robot. The routing for each batch will be determined based on the assigned items and the robot's location. For example, items 1,5,4,6 from orders 1 and 3 are assigned to robot 1.

The objective of this work is to find the best order allocation to minimize the total distance traveled and congestion by autonomous picking carts while satisfying the capacity and sequencing constraints. To achieve this, we propose five different mathematical models and find that the generalized quadratic assignment model performs the best. Since the generalized quadratic assignment model is an NP-hard problem, we proposed an iterative Simulated Annealing (SA) algorithm to solve it, and we used Lagrangian Relaxation to find lower bounds.

An open-source dataset has been used to evaluate the proposed models and approach. Based on the comparison of total travel distance among all models, it can be seen that SA outperformed all models. SA can find a solution very close to the lower Lagrangian bound, and it is much faster than all other models while solving large instances. The contributions of this work are threefold. First, we model the order batching problem in an automated warehouse with the objective of minimizing travel distance and congestion. Second, we propose an iterative SA algorithm that generates an initial solution by using the K-means clustering algorithm. Third, we carry out testing and numerical analysis on an open-source dataset to demonstrate the efficiency of the proposed approach.

This thesis is organized as follows. Chapter 2 presents an overview of the literature related to the problem. In Chapter 3, we describe the Order Batching Problem (OBP), provide mathematical formulations, and propose the SA algorithm and Lagrangian relaxation. We evaluate the performance of the proposed models using open source data in

Chapter 4. Finally, we present some concluding remarks and future research directions in Chapter 5.

Chapter 2

Literature Review

The increasing need for automation technology has prompted extensive research into diverse applications of warehouse automation for material handling systems. Zheng Zhang (2023), for example, conducts a comprehensive analysis of automation products in the field of warehousing, specifically targeting e-commerce companies. The primary emphasis of the study centers around operational-level concerns, encompassing topics such as order allocation, task scheduling, rack allocation, and robot path planning. The literature review aims to unfold the challenges associated with robotic mobile fulfillment systems and the proposed potential solutions. Storage location assignment will be addressed initially, followed by the latest developments in warehouse fulfillment systems for autonomous mobile robots, and finally, the order batching problem.

2.1 Storage Location Assignment

The implementation of Robotic Process Automation (RPA) has been found to enhance customer satisfaction due to its heightened reliability, efficiency, adherence to regulatory requirements, and reduced susceptibility to human errors. The optimization of warehouse layout plays a vital role in minimizing order fulfillment durations and enhancing overall warehouse efficiency. A storage assignment strategy refers to the method by which items are distributed among storage locations and the subsequent impact on the retrieval time of the storage system (Roodbergen and Vis, 2009). There exist multiple techniques for the allocation of products within rack storage facilities. Hausman et al. (1976) identifies five distinct storage assignment techniques. The methods encompassed in this study include dedicated storage assignments, random storage assignments, closest open location storage assignments, full-turnover-based storage assignments, and class-based storage assignments (Zone clustering).

The storage assignment method, or dedicated storage, assigns each product to a predetermined and unchanging location. This particular approach necessitates a significant amount of physical area. The random storage assignment method is capable of assigning products to any vacant location with equal probabilities. In the context of assigning storage locations, the product will be allocated to the nearest available location in terms of proximity. This approach ensures that all products will be strategically positioned at the forefront of the warehouse, in close proximity to the input/output points. Consequently, the rear section of the warehouse will be left vacant. The storage assignment method known as full-turnover-based determines the location of a product by considering its demand frequencies. Items that occur most frequently will be located in close proximity to the input/output (I/O) points. The storage assignment method based on class is commonly referred to as zone clustering. The survey paper by Roodbergen and Vis (2009) provides a comprehensive overview of storage assignment methods and can serve as a valuable resource for obtaining in-depth explanations on the topic.

Zone clustering is a storage assignment model that falls within the realm of storage management techniques. Implementing zone-based assignments rather than random assignments can potentially minimize the overall travel time of robots effectively. Implementing a synchronized zone order picking system can potentially enhance warehouse efficiency (Roy et al., 2019). The primary objective of zone clustering is to minimize overall travel expenses. Several researchers have directed their attention toward the issue of zone clustering, employing various solution approaches.

Kim et al. (2020) investigated the problem of item assignment in the RMFS with the objective of maximizing the sum of the similarity values of the items in each rack. Nevertheless, the current methodology has yet to be extended to include the integration of zone clustering into the RMFS. The study conducted by Li et al. (2020a) centers around the assessment methodology for the storage location assignment policy, specifically emphasizing energy consumption awareness. The current study employs a novel storage assignment method whereby mobile racks with varying turnover rates are dispatched to the autonomous storage area. Within a warehouse setting that contains a range of scales, the suggested methodology has the potential to optimize both order-picking efficiency and the utilization of Automated Guided Vehicle (AGV) energy to varying extents. The results indicate that the implementation of the new strategy leads to a significant improvement in order-picking efficiency and a reduction in energy consumption by AGVs. Roy et al. (2019) conducted a study on the zone assignment algorithms used in the zone clustering problem within the single-deep scenario. Furthermore, a two-stage stochastic model was developed and utilized to investigate the allocation strategies of robots across multiple storage zones, employing multi-class closed queueing network models. Lamballais et al. (2016) achieved optimization of various parameters in the RMFS by introducing a novel semi-open queueing network. These parameters include the maximum number of mobile racks per SKU, the ratio of pick stations to replenishment stations, and the replenishment level per rack.

The work of Keung et al. (2020), Keung et al. (2019), and Lee et al. (2019) developed a cloud-based framework for Cyber-Physical Systems (CPS) that addresses collision avoidance, conflict resolution, and charge scheduling. However, it is important to note that this framework has not yet gained recognition as a RPA. In order to mitigate human errors, it is possible to employ a software robot that is capable of executing a multitude of tasks. In order to mitigate the overall operating expenses within RMFS, it is imperative to consider implementing a data-driven approach, specifically for addressing the storage location assignment problem and the order classification problem. The utilization of the data mining approach facilitates the process of pattern discovery and enables the simplification of prediction procedures. Keung et al. (2021) conducted a comparative analysis of nine distinct clustering algorithms, encompassing both semi-supervised and unsupervised approaches. These algorithms, such as K-means, Gaussian Mixture Models, and Bayesian Gaussian Mixture Models, were evaluated in terms of their performance in zone clustering within the RMFS context. The results indicated that such algorithms achieved an average accuracy rate of 95%.

Lamballais et al. (2016) investigate the impact of workstation placement in close proximity to the storage area on the overall order throughput capacity. Yuan and Gong (2017) propose the utilization of an open queueing network to effectively determine the optimal quantity of robots and their corresponding average speed required to attain a desired throughput time. Zou et al. (2017) employ a neighborhood search approach to identify an assignment rule that closely approximates the ideal. They propose a workstation assignment rule that considers the workstations' handling speeds. Furthermore, the findings of this study indicate that the handling-speeds-based assignment rule outperforms the random assignment approach in situations where there is a substantial discrepancy in handling time among workers. Wu et al. (2020) develope a queue network model in order to assess the performance of RMFS. The proposal suggests relocating the picking stations within the storage area, resulting in the design of seven layout scenarios based on the arrangement of the stations and the storage area. The findings demonstrate a substantial performance enhancement as a result of vertical zoning. Numerous studies have demonstrated that the allocation of storage is a critical factor in enhancing order-picking efficiency within RMFS.

2.2 Autonomous Mobile Robot Fulfillment Systems

The mobile fulfillment system (MFS), which utilizes mobile-rack technology, was originally developed by Kiva Systems, a subsidiary of Amazon Robotics (Boysen et al., 2018). Amazon has developed a logistical model called "Shelf to The People" using the "Kiva System," which has significantly influenced the evolution of the picking process. Amazon's acquisition of the Kiva system took place in 2012, followed by its implementation in August 2015. The Kiva Robots are widely recognized for their ability to improve workflow efficiency, despite requiring human labor for operation. The Kiva Robots exclusively complement human labor and do not autonomously replace human positions. Hence, using Kiva Robots can effectively reduce the time pickers would have otherwise expended on walking. In order to fulfill their responsibilities, it is necessary for Amazon employees to be physically present at a designated location. The order-picking process involves the utilization of robots to transport mobile racks of shelves alongside static pickers who are responsible for selecting the items required to fulfill customers' orders. The research conducted on mobile fulfillment systems for goods-to-person operations was primarily based on the Kiva system. In essence, implementing an automated mobile robot solution for transporting goods to individuals poses several captivating decision-making challenges, thereby providing inspiration for numerous research endeavors.

Automated Mobile Robots (AMRs) are utilized to transport storage shelves to designated pick stations, where operators engage in the process of selecting products within mobile shelf-based order-picking systems. AMRs align themselves in a sequential formation and patiently queue at designated picking stations upon their arrival. The primary element of AMR solutions for goods-to-person operations is the queueing behavior, as stated by Enright and Wurman (2011). One potential approach to enhancing productivity and increasing the rate of picking operations is to effectively manage the queue of activities at the stations, thereby ensuring the continuous engagement of the pickers Wurman et al. (2007). Conversely, the queues significantly impact various aspects of the robotic solution, such as the order cycle time, AMR productivity, and the required number of robots. Queuing theory models are widely regarded as the fundamental basis for the majority of studies conducted on goods-to-person AMR systems, owing to the influence of these factors.

The distribution of items within each Stock Keeping Unit (SKU) spreads throughout the racks. Consequently, selecting optimal racks and determining their arrival sequence are interdependent decisions that significantly impact the order sequencing decision. A close and mutually limiting relationship characterizes the interconnection between orders, racks, and workstations (Yang et al., 2021). The order sequence for a single workstation and the interdependent rack sequence in the Kiva system was determined by Boysen et al. (2017). The researchers provided decomposing techniques utilizing simulated annealing to tackle the two subproblems in isolation effectively. Nevertheless, the prolonged delay was disregarded in order to ensure the uninterrupted fulfillment of customer orders by the current rack system. The put-to-light system successfully addressed the aforementioned issue, as demonstrated by Füßler and Boysen (2017). This study involves the categorization of products based on their SKUs into a diverse bin, necessitating the coordination of workbench deliveries and ongoing order batches. Valle and Beasley (2021a) introduces two heuristic modes that incorporate partial integer optimization. This study represents the inaugural contribution in the existing body of literature by introducing an optimization approach that addresses the concurrent allocation of orders and racks among multiple pickers.

The scalability and non-modification requirements of collaborative AMR solutions have received more attention in comparison to shelf-based goods-to-person solutions in recent times Meller et al. (2018). Collaborative Meet-In-Aisle (MIA) solutions have garnered significant attention in the realm of e-commerce fulfillment facilities as an alternative option. Mobile autonomous mobile robots (MIA AMRs) possess the capability to navigate within the designated picking area and retrieve orders. The AMR is provided with a set of objects to retrieve in the given scenario and proceeds toward the initial destination. Upon the arrival of the AMR at the designated pick location, it promptly provides the picker with explicit instructions to select the necessary products carefully and subsequently place them within the specifically assigned container located on the AMR. Once the task is completed, the AMR proceeds to the subsequent picking location in order to fulfill its assigned tasks. These robotic devices enhance the efficiency of order-picking operations through the utilization of intelligent processors integrated into their onboard systems. In essence, the robots possess the capability to accurately determine positions, strategically organize selections, and provide assistance to pickers during the decision-making process, all facilitated by the integrated on-screen technology. According to Ghelichi and Kilaru (2021), it has been observed that MIA AMRs frequently possess containers, thereby eliminating the need for workers to push carts manually. The ability to direct pickers and transfer goods substantially reduced the training expenses associated with workers in fulfillment centers. In conclusion, the utilization of MIA AMR systems presents a substantial potential for augmenting order-picking operations within e-commerce fulfillment centers across various aspects Ghelichi and Kilaru (2021). One of the primary shortcomings of MIA AMR solutions pertains to their inherent constraint in facilitating the transportation of sizable objects.

The literature on collaborative solutions is narrow. There are only a few papers on this topic since most researchers worked on shelf-based goods-to-person systems. Ghelichi and Kilaru (2021) developed analytical models for Last-Mile Delivery (LMD) and MIA mobile solutions.

2.3 Order Batching

Order picking is a crucial process in warehouse management, contributing significantly to overall cost and time. Minimizing pickers' travel distances is one of the biggest order-picking issues because it directly affects the operation's productivity and efficiency. The joint order batching and picker routing problem (JOBPRP) aims to assign orders to picker routes and batches simultaneously. The JOBPRP has received a lot of scientific interest recently, which has resulted in the development of numerous approaches and algorithms.

In manual picker-to-parts warehouses, Kübler et al. (2020) proposed a new iterative strategy for resolving the joint dynamic storage location assignment, order batching, and picker routing problems. The method employs a two-stage heuristic approach, with the first stage taking dynamic storage location assignment and order batching into account and the second stage taking picker routing into account. In the DSLA-OB-PR problem, picking routes for pickers to get the goods for each batch of orders, batching of orders, and assigning storage locations to items are all factors. The challenge is satisfying the order limitations while minimizing the pickers' distance traveled. To deal with the DSLA-OB-PR problem, the proposed method takes a two-step approach. In the first step, an initial solution is generated using a heuristic algorithm that successively assigns goods to storage locations, batches the orders, and assigns pickers to the batches. In the second step, an iterative method is used to refine the initial solution by iteratively updating the storage assignments, order batches, and picker routing assignments to lower the overall journey distance. The experimental results reveal that the suggested approach surpasses existing methods in terms of solution quality and computing efficiency. The suggested technique also displays robustness in dealing with dynamic conditions such as changes in order profiles and picker availability.

Arbex Valle and Beasley (2020) present an order batching estimate approach that considers picker distance. The approach groups orders based on their location and size using a clustering algorithm and then employs a heuristic to identify the ideal sequence of orders within each cluster. The authors compare the suggested method to previous ways and analyze it on a set of test situations, proving that it can drastically cut trip distance. The suggested technique initially estimates the distance between each pair of storage locations using the Euclidean distance formula. The distance matrix is then used to calculate the distance traveled by pickers for various batch sizes. The calculation presented in this study enables the determination of the optimal batch size, which aims to minimize the overall distance covered by the pickers. The method under consideration incorporated a substantial set of constraints, which were introduced into the formulation with the aim of enhancing the efficacy of linear programming relaxation. The study conducts a comparative analysis between the proposed method and two alternative methods: the nearest neighbor heuristic and the largest gap heuristic. The comparison is made in relation to both solution quality and computational efficiency. The experimental findings demonstrate that the proposed method exhibits superior performance compared to alternative methods, specifically concerning the aggregate distance covered by pickers when considering a predetermined set of orders.

A novel approach was proposed by Aerts et al. (2021) to address the JOBPRP in manual picker-to-parts warehouses. The JOBPRP entails the identification of the most efficient order batches and picker routes, with the aim of minimizing the overall distance covered by pickers while still adhering to the constraints imposed by the orders. This study employs a clustered vehicle routing problem (CVRP) framework to model the JOBPRP. Each cluster corresponds to a batch of orders in this framework, and the pickup and delivery tasks are represented as a vehicle routing problem. The present study introduces a heuristic algorithm that employs a two-phase approach for addressing the CVRP. During the initial phase, a solution is derived using a two-step approach: clustering the orders first and then constructing routes for each cluster. During the second phase, a local search algorithm is utilized to enhance the initial solution by iteratively exchanging orders among the clusters and re-optimizing the routes. This study conducts a comparative analysis between the proposed algorithm and established methods, namely the nearest neighbor heuristic and the largest gap heuristic, with regard to both solution quality and computational efficiency. The experimental findings demonstrate that the algorithm proposed in this study surpasses alternative methods in both solution quality and computational efficiency. Furthermore, the present study conducts a sensitivity analysis of the suggested algorithm, examining its response to various factors, including the number of orders, the number of pickers, and the dimensions of the warehouse. The analysis demonstrates that the algorithm proposed exhibits robustness and scalability, enabling it to effectively manage dynamic scenarios, including fluctuations in order profiles and picker availability.

Cergibozan and Tasan (2022) propose a methodology that utilizes a genetic algorithm to address the order batching problem. The method proposed in this study employs a hybrid encoding scheme that integrates both binary and permutation representations of orders. Additionally, a novel mutation operator is introduced, which takes into account the pick path of orders. The authors implement the proposed method in a case study conducted at a distribution center, wherein they demonstrate its ability to yield substantial enhancements in the overall travel distance.

Xie et al. (2022) present a mathematical model that addresses the integrated order batching and routing problem within the context of multi-depot AGV-assisted mixed-shelves warehouses. The authors posit that current models addressing the joint order batching and picker routing problem lack consideration for the integration of automated guided vehicles (AGVs), a technology that holds the potential for substantially enhancing the efficiency of order-picking operations within warehouse environments. The proposed model integrates the utilization of AGVs by taking into account factors such as AGV availability, capacity, and routing. The model also incorporates a mixed-shelves feature, which facilitates the storage of various products within a single location. This effectively minimizes the number of locations that pickers are required to visit. The authors substantiate the proposed model's validity through a case study involving a mixed-shelves warehouse with multiple depots where AGVs are utilized. Furthermore, they conduct a comparative analysis of its performance against existing models that do not incorporate AGVs. The findings indicate that the model exhibits a notable superiority over current models in relation to order-picking efficiency and warehouse utilization. This paper makes a significant scholarly contribution to the existing body of literature on the joint order batching and picker routing problem. It specifically focuses on the integration of AGVs into the order-picking processes within warehouse operations. This statement underscores the potential advantages associated with the utilization of AGVs in enhancing the efficacy of order-picking operations and mitigating expenses within warehouse environments.

Matusiak et al. (2014) introduced a methodology comprising of two distinct sub-algorithms. The routing process utilizes an optimal A* algorithm, while the batching process employs a simulated annealing algorithm. The latter algorithm estimates the potential savings achieved by combining more than two customer orders into a single batch, thereby avoiding unnecessary routing. The primary objective of the PCES algorithm is to reduce the computational complexity associated with batch evaluations in the context of warehouse optimization. The results indicate that the proposed batching heuristics exhibit competitive performance in terms of solution time and quality compared to other state-of-the-art methods. The utilization of the REMIX heuristic, which integrates stochastic elements, enables PCES to exhibit adaptability and attain favorable outcomes. The algorithm demonstrates superior performance in handling wave sizes of medium to large magnitudes and exhibits improved solution outcomes compared to the C&W(ii) algorithm when processing batches consisting of three customer orders. However, it has been observed that for wave sizes that are relatively small, other heuristics exhibit superior performance compared to PCES. This can be attributed to the inconsistent behavior of PCES within this particular range. The performance of PCES is relatively satisfactory in comparison to optimal batching and routing methods for small wave sizes, as it manages to achieve travel distance reductions while requiring less computational time. Furthermore, it facilitates the assessment of various batch sizes within any given warehouse that adheres to precedence-constrained routing.

In summary, the literature review provides an overview of the recent developments in the field of order batching and picker routing within warehousing. This encompasses a range of techniques such as heuristics, metaheuristics, and exact methods. The papers under review provide evidence of the efficacy of clustering, genetic algorithms, and dynamic programming in addressing the challenges posed by the JOBPRP, thereby enhancing order-picking efficiency and productivity. Moreover, the literature review highlights the significance of taking into account dynamic variables, such as fluctuating order profiles and the availability of pickers, when optimizing the processes of order batching and picker routing.

It is important to acknowledge that the reviewed papers exhibit certain limitations and offer opportunities for future research. For example, the majority of the proposed methodologies are assessed using test instances of limited scale, which may limit their applicability to real-world warehouses characterized by a substantial volume of orders and pickers. Additionally, certain methods that have been suggested may exhibit high computational demands, rendering them unsuitable for the purposes of online or real-time optimization of order picking. Hence, future investigations may prioritize the development of more scalable and efficient methodologies capable of managing dynamic and extensive order-picking operations.

In general, the papers under review offer significant insights into the issue of order batching and picker routing in warehousing, presenting practical solutions that have the potential to improve order-picking efficiency and productivity. The implementation of these solutions has the potential to yield significant advantages for warehouse managers and operators. These benefits include reducing operational costs, enhancing customer satisfaction, and attaining a competitive edge within the market.

Chapter 3

Problem Definition and Solution Methodology

In this study, we consider Meet-In-Aisle autonomous order-picking solutions under a class-based storage policy, meaning that our warehouse will be divided into classes based on frequency. The goal is to minimize the total distance traveled by minimizing the total pairwise distance of all items assigned to the same batch and, implicitly, the route congestion measured by intersections between robot travel paths.

The system that is used in this study has some assumptions. We consider a class-based storage policy, which means that products occupy specific zones in the warehouse based on certain criteria, such as usage frequency. Operators are assigned to zones and can only travel within their own zones. A zone has multiple aisles. An operator can be responsible for one or multiple aisles. In order to navigate efficiently, robots are required to travel the aisles within designated zones. Upon entering an aisle, a robot must go through the entire length of the aisle. The operators will travel from robot to robot in their designated aisles.

Robots' velocity is constant. Acceleration and deceleration are assumed to be negligible. The robots will collect a series of items from the assigned order batch. There is only one order consolidation and workstation in the warehouse where robots bring back collected items to pack orders. Orders are assigned to batches to create a pick list for the robots. The congestion is taken into account based on the intersection of robot collection routes.

A sample layout of a warehouse that is used in this study is given in Figure 3.1 in which there are 20 aisles between the first rack and the last rack on one side of the warehouse and 20 aisles on another side. The distance of aisles is 3 meters, which can allow robots to travel without collusion. The front side is dedicated to order consolidation and the robots' waiting area.

The order allocation problem we consider focuses on assigning items to batches and batches to robots. Since we know the location of each item in the warehouse, we use their location to calculate our parameters. Details are provided in the next section.

3.1 Mathematical Models

Given a set of customer orders with a list of items to be picked and their corresponding storage locations, the goal of the robotic order batching problem is to group the orders into batches in a way that minimizes the total distance traveled by robots and congestion of robots when retrieving items from storage locations. Congestion is an important factor to consider in designing and operating automated warehouses, as it can significantly impact system performance and efficiency. Congestion in automated warehouses can occur when a high volume of products is transported, processed, or stored, leading to bottlenecks, delays, and reduced throughput. This can result in longer order processing times, increased waiting times, and reduced overall productivity of the warehouse. Several studies have highlighted

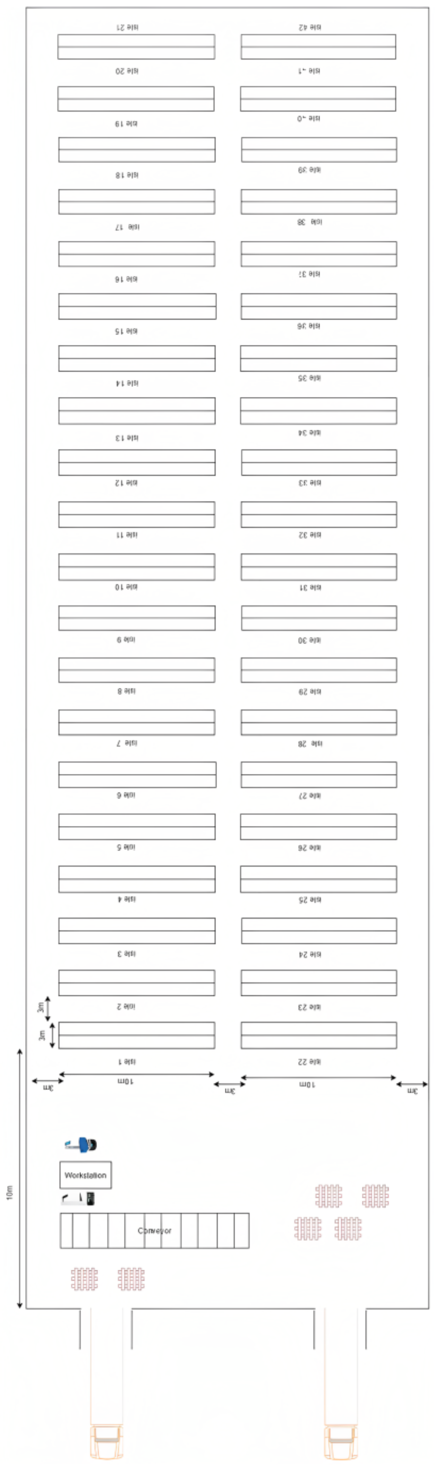

Figure 3.1: Layout of the warehouse

the importance of congestion management in automated warehouses. For example, in a study by Hsu et al. (2005), the authors demonstrated that congestion in the picking area of a warehouse could result in reduced throughput and increased order processing times.

The robotic order batching problem takes into account various constraints such as storage capacity, item availability, order deadlines, and order size. Formally, let $O = \{1, 2, ..., n\}$ be the set customer orders, where each order consists of a list of m items to be picked and their corresponding storage locations, represented as $(item_1, location_1)$, $(item_2, location_2)$, ..., $(item_m, location_m)$. The warehouse has a set of robots that can collect items from storage locations and deliver them to a designated station for packing and shipping. Let $I = \{1, ..., n_I\}$ be the set of items that need to be collected to complete orders. Indices $i, j \in I$ will be used for items.

The goal is to partition the items into b batches $B = \{1, ..., n_B\}$, where each batch consists of a set of items that can be picked together in a single trip, such that the total distance traveled by the robots or operators and the congestion are minimized. We use indices $b \in B$ for batches. The total traveled distance is the sum of the rectilinear distances of items in each batch in the order of close to furthest. The distance between two items is defined as d_{ij}, and it is calculated as the minimum rectilinear distance of two items based on the location of aisles and racks. Let $K = \{1, ..., n_K\}$ be the set of aisles in the warehouse, indexed by k. While forming the batches, each robot's number of visited aisles is essential to reduce the travel distance since robots move in one direction through aisles.

The problem takes into account various constraints, such as the storage capacity of the robots or operators, the size of the orders, and the number of visited aisles by robots. The solution should ensure that all the orders are fulfilled and the capacity of the robots or operators is not exceeded.

There are some parameters that are required to model the problem. Since the storage location of each item is known, the binary parameter a_i^k is defined to represent the items' aisle locations, i.e., a_i^k is 1 if item $i \in I$ is on aisle $k \in K$, 0 otherwise. Each robot has a container that can host a certain number of items, so Q represents the maximum capacity of a container or robot. Orders contain different items with various amounts captured by parameter m_i, which is the total number of items in the given order. Finally, there is a limited number of available robots in the warehouse, so the number of batches cannot exceed n_r.

In order to model the robotic order batching problem, we define three sets of binary decision variables. We define the binary variables x_{ib} that take value 1 if item $i \in I$ is in batch $b \in B$, z_{kb} that take value 1 if batch $b \in B$ contains items in aisle $k \in K$, y_b which take value 1 if batch b is formed.

Next, we provide different models for the robotic order batching problem with congestion. The first formulation is:

$$
\begin{aligned}
[OBP1] : \min \quad & \sum_{k \in K} \sum_{b \in B} z_{kb} && && (3.1)\\
\text{s.t.} \quad & \sum_{b \in B} x_{ib} = 1 && \forall i \in I && (3.2)\\
& \sum_{i \in I} a_i^k x_{ib} \leq z_{kb} && \forall b \in B, \forall k \in K && (3.3)\\
& \sum_{i \in I} m_i x_{ib} \leq Q y_b && \forall b \in B && (3.4)\\
& \sum_{b \in B} y_b \leq n_r && && (3.5)\\
& x_{ib}, z_{kb}, y_b \in \{0, 1\} && \forall b \in B, \forall k \in K, \forall i \in I && (3.6)
\end{aligned}
$$

The objective function 3.1 minimizes the total number of aisles visited by robots. Constraints 3.2 make sure each item is assigned to exactly one batch. Constraints 3.3 are linking constraints for decision variables that ensure that if an item is assigned to a batch, that batch will be assigned to the aisle of the item. Constraints 3.4 make sure that batch capacity is not exceeded. Constraint 3.5 states that the total number of batches cannot exceed the number of robots.

Instead of minimizing the total number of visited aisles, we can minimize the maximum number of aisles visited by any batch, leading to:

$$[OBP2] : \min \quad z \tag{3.7}$$

$$\text{s.t.} \quad z \geq \sum_{k \in K} z_{kb} \qquad \forall b \in B \tag{3.8}$$

$$(3.2), (3.3), (3.4), (3.5), (3.6).$$

[OBP1] and [OBP2] only consider the number of visited aisles. Since we do not include the maximum distance between assigned two items, a robot may travel from the first aisle to the last one. To solve this problem, we add maximum distance constraints to our model. Let w be the maximum distance between two assigned items for a batch. Since z and w are not on the same scale, we multiply z with a weight factor $\frac{L+W}{K}$ where L is the length of the warehouse, W is the width of the warehouse, and K is the total number of aisles. The resulting model is:

$$[OBP3] : \min \quad \frac{L+W}{K} z + w$$

$$\text{s.t.} \quad z \geq \sum_{k \in K} z_{kb} \qquad \forall b \in B \qquad (3.8)$$

$$(3.2), (3.3), (3.4), (3.5), (3.6)$$

$$w \geq d_{ij} x_{ib} x_{jb} \qquad \forall i \in I, \forall j \in I, \forall b \in B \tag{3.9}$$

$$w \geq 0.$$

Another alternative is to focus only on the maximum distance between assigned items because this may have a higher impact on reducing the travel distance. This is achieved by removing the z_{kb} variables and constraints 3.3. Leading to:

$$
\begin{aligned}
[OBP4]: \min \quad & w \\
\text{s.t.} \quad & \sum_{b\in B} x_{ib} = 1 && \forall i \in I && (3.2) \\
& \sum_{i\in I} m_i x_{ib} \le Q y_b && \forall b \in B && (3.4) \\
& \sum_{b\in B} y_b \le n_r && && (3.5) \\
& w \ge d_{ij} x_{ib} x_{jb} && \forall i \in I, \forall j \in I, \forall b \in B && (3.9) \\
& w \ge 0 && && \\
& x_{ib}, z_{kb}, y_b \in \{0,1\} && \forall b \in B, \forall k \in K, \forall i \in I. &&
\end{aligned}
$$

A closely related model to [OBP4], is the generalized quadratic assignment model:

$$
\begin{aligned}
[OBP5]: \min \quad & \sum_{i\in I} \sum_{j\in I} d_{ij} \sum_{b\in B} (x_{ib} x_{jb}) \\
\text{st.} \quad & \sum_{b\in B} x_{ib} = 1 && \forall i \in I && (3.2) \\
& \sum_{i\in I} m_i x_{ib} \le Q y_b && \forall b \in B && (3.4) \\
& \sum_{b\in B} y_b \le n_r && && (3.5) \\
& x_{ib}, y_b \in \{0,1\} && \forall b \in B, \forall i \in I && (3.6)
\end{aligned}
$$

The Generalised Quadratic Assignment Problem models a wide range of problems that seek to minimize the overall cost of pairwise interactions among entities, such as equipment or tasks, and where the placement of these entities into potential destinations is restricted by the capacity of available resources. Being NP-hard, the generalized quadratic assignment problem is difficult to solve directly. Hence, we propose Lagrangian relaxation and metaheuristics to solve [OBP4] and [OBP5].

Each model is evaluated using a real-world dataset to determine its performance. During initial testing, it was observed that [OBP1] and [OBP2] do not adequately reduce travel distance, with routing assignments often covering long distances. In order to minimize the travel distance required for each robot to collect all assigned items, it is expected to have assigned items situated in close proximity to one another. Hence, it is expected that visible clusters of items are observed. The problem at hand is effectively addressed by [OBP3] algorithm, whose solution enables the formation of clusters.

The following figures, Figure 3.2 to 3.6, display solutions from each model where rectangles represent items and their location in the warehouse. To better identify the clusters, circles are used to denote the hypothetical center of the cluster, with spokes linking to the location of the items. A small example involving 5 orders and 5 batches is used. A good solution should contain distinctive clusters, thus reducing congestion by eliminating intersections between robot travel paths.

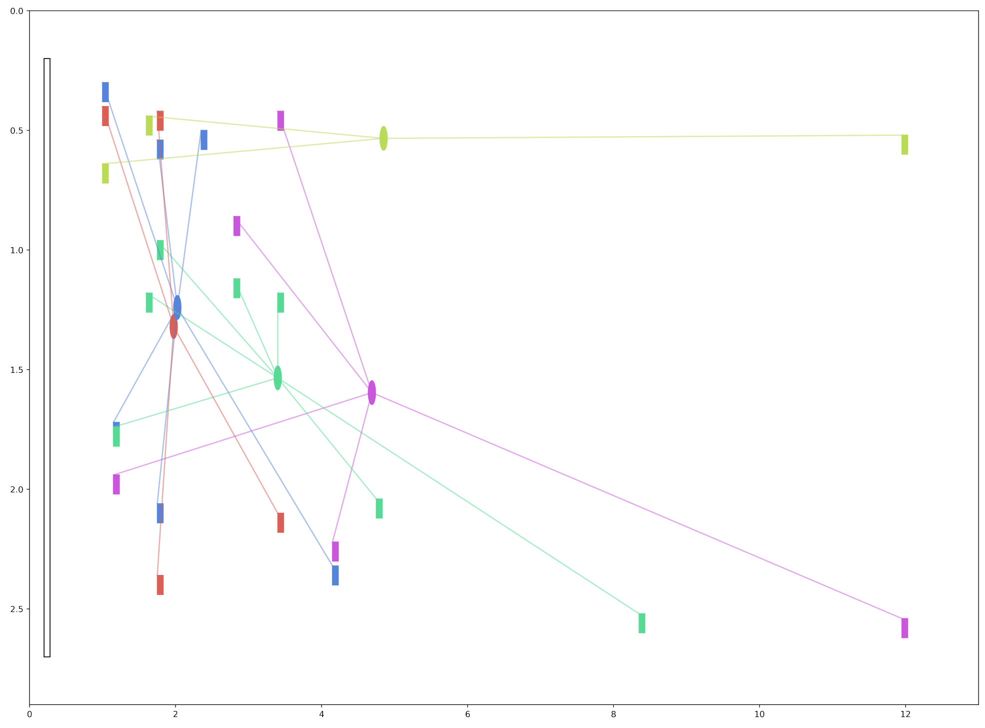

Figure 3.2: Assigned items' distance from the center of batches for [OBP1]

Figure 3.2 displays the solution for [OBP1]. As observed, robots are required to travel from the front to the back to gather items as [OBP1] does not minimize the distance traveled by each individual robot. Instead, it focuses on minimizing the number of visited aisles. Consequently, the solution results in paths of robots crossing each other, leading to potential congestion.

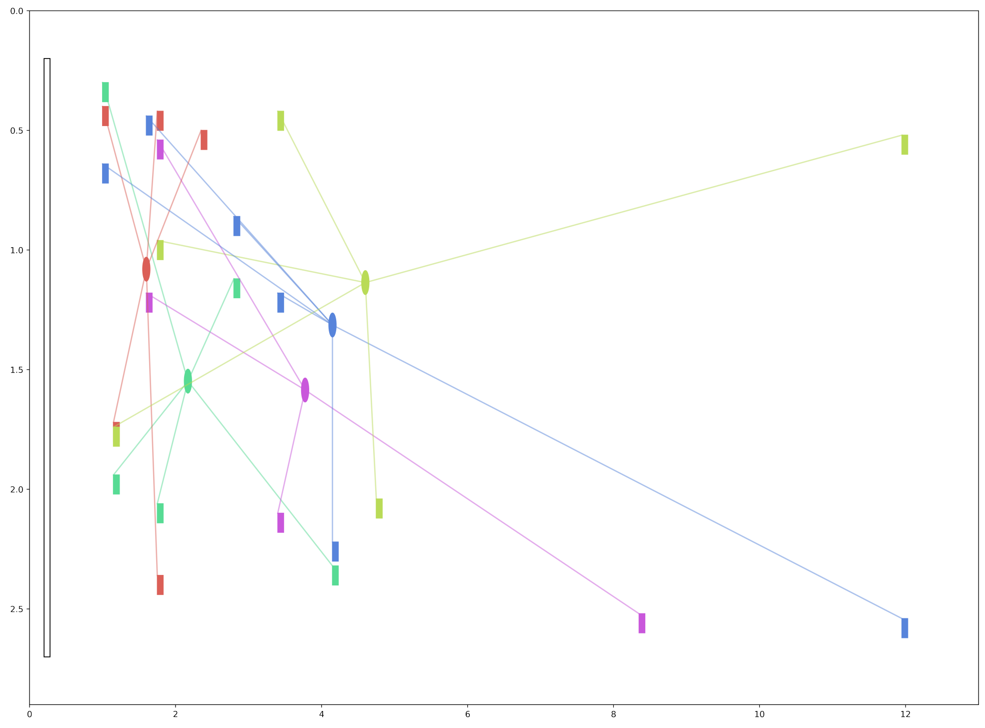

Figure 3.3: Assigned items' distance from the center of batches for [OBP2]

Figure 3.3 displays the solution for [OBP2] and reveals that the issue observed in [OBP1] persists,i.e., congestion due to the intersection of multiple paths.

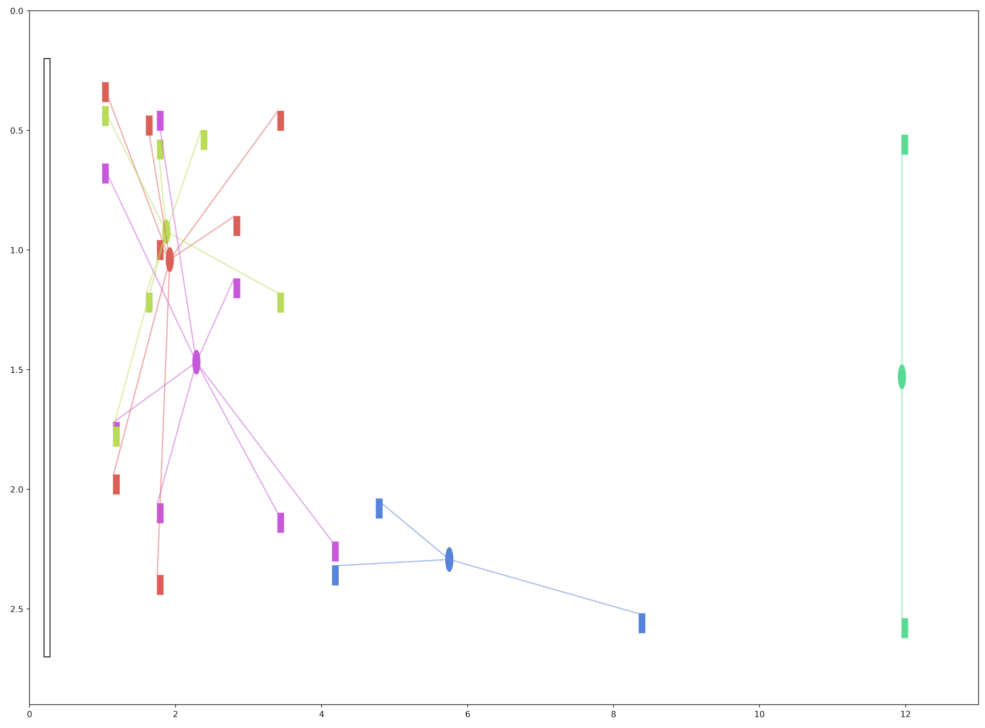

Figure 3.4: Assigned items' distance from the center of batches for [OBP3]

Figure 3.4 displays the solution for [OBP3]. As observed, [OBP3] exhibits superior cluster formation capabilities compared to [OBP1] and [OBP2]. This can be attributed to the fact that [OBP3] takes into account the distances between assigned items. [OBP3], however, is computationally demanding. Only small instances can be solved to optimality.

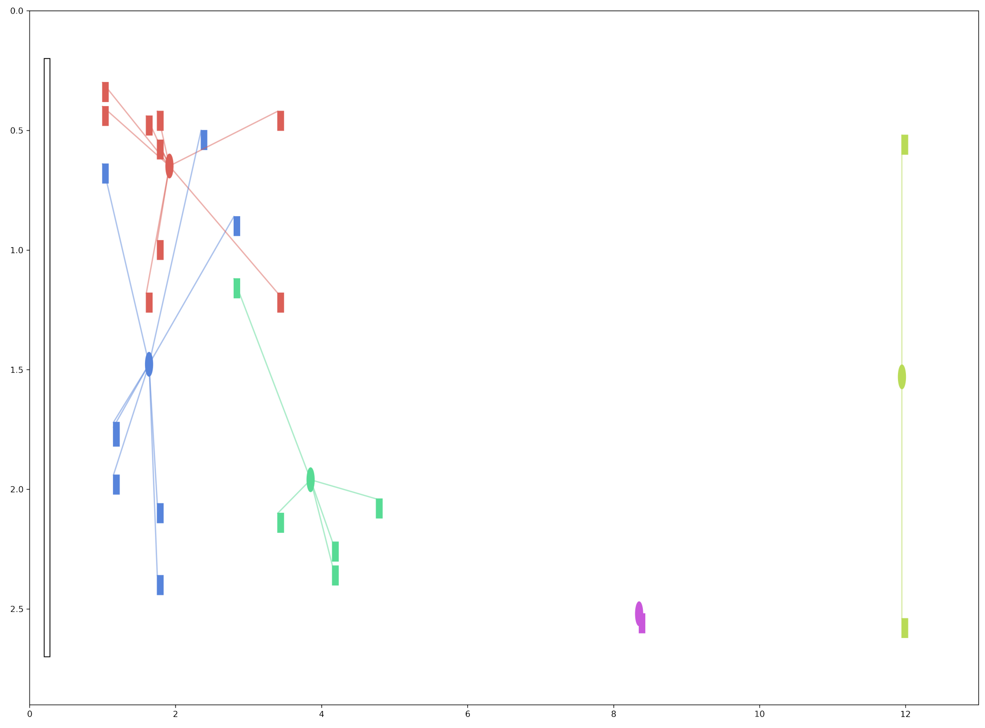

Figure 3.5: Assigned items' distance from the center of batches for [OBP4]

Figure 3.5 displays the solution for [OBP4] where it is evident that [OBP4] effectively minimizes the distance between assigned items. Moreover, it should be noted that [OBP4] exhibits a lower number of decision variables compared to [OBP3] enhancement in terms of more optimal pathways.

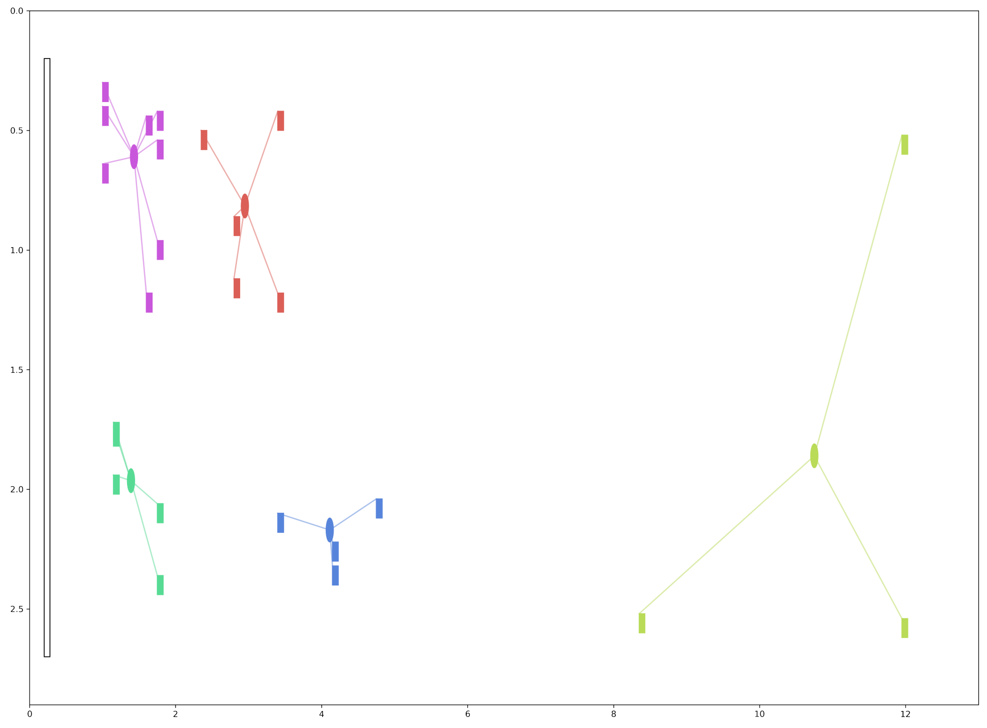

Figure 3.6: Assigned items' distance from the center of batches for [OBP5]

Figure 3.6 displays the solution for [OBP5] where it is illustrated that it exhibits superior performance compared to [OBP4], specifically in terms of item clustering. Similar to [OBP4], [OBP5] is computationally intractable for large instances. It is, however, more prone to decomposition, such as Lagrangian relaxation.

Table 3.1 presents a comparison of the computational time and objective value for the five models with varying instance sizes. It displays the number of orders (Order), the count of unique Stock Keeping Units (SKUs) (Item), the total quantity of collected SKUs

(Quantity), the number of batches (Batch), and the capacity of each robot. A time limit of 900 seconds is imposed, and the objective of the best feasible solution obtained is provided. As [OBP5] is the most time-consuming, we proposed to decompose it using Lagrangian relaxation.

3.2 Lagrangian Relaxation for OBP5

We relax constraints 3.2 and 3.5 with Lagrange multiplier λ_i, and $\mu \geq 0$, respectively. The resulting subproblem is:

$$\begin{aligned} \min \quad & \sum_{i \in I} \sum_{j \in I} d_{ij} \sum_{b \in B} (x_{ib} x_{jb}) - \sum_{i \in I} \sum_{b \in B} \lambda_i x_{ib} + \sum_{i \in I} \lambda_i + \mu \sum_{b \in B} y_b - n_r \mu \\ \text{s.t.} \quad & (3.4), (3.6). \end{aligned}$$

It decomposes into n_b subproblems, one for each batch:

$$\begin{aligned} \min \quad & \sum_{i \in I} \sum_{j \in I} d_{ij} (x_i x_j) - \sum_{i \in I} \lambda_i x_i + \mu y \\ \text{s.t.} \quad & \sum_{i \in I} m_i x_i \leq Q y. \end{aligned}$$

					OBP1		OBP2		OBP3		OBP4		OBP5	
Order	Item	Quantity	Batch	Capacity	Optimal value (# aisles)	Time (s)	Optimal value (# aisles)	Time (s)	Optimal value	Time (s)	Optimal value (m)	Time (s)	Optimal value (m)	Time (s)
5	25	173	5	100	24	0.08	5	0.09	77.6	1.45	32.3	0.98	1365.40	24.42
6	45	734	10	100	44	0.13	5	0.14	81.6	22.99	33.7	22.56	1737*	900
7	46	814	10	100	45	0.11	5	0.17	92.9	35.10	44.0	19.91	2059.9*	900
10	65	1026	15	100	64	0.21	5	0.22	92.9	71.59	20.2	486.16	1968.8*	900
12	83	1383	20	200	82	0.26	5	0.32	92.9	239.12	11.9	376.71	1151.6*	900
15	105	2120	25	400	104	0.37	5	0.45	148.2*	900.00	18.3*	900	1018.8*	900

* best found feasible solution.

Table 3.1: Comparing optimal values of different order and item numbers for all models

Which is a quadratic knapsack problem that can be linearized using the new decision variable $v_{ij} = x_i x_j$.

$$\begin{aligned}
\min \quad & \sum_{i\in I}\sum_{j\in I} d_{ij}(v_{ij}) - \sum_{i\in I} \lambda_i x_i + \mu y & \\
\text{s.t.} \quad & \sum_{i\in I} m_i x_i \leq Qy & \\
& v_{ij} \leq x_i & \forall i \in I, \forall j \in I \\
& v_{ij} \leq x_j & \forall i \in I, \forall j \in I \\
& v_{ij} \geq x_i + x_j - 1 & \forall i \in I, \forall j \in I \\
& x_i, v_{ij} \in \{0,1\} & \forall i, j \in I.
\end{aligned}$$

Possible solutions for subproblems can be either $y = 0$ with x_i=0 or $y = 1$ with its corresponding x_i being to the solution to

$$\begin{aligned}
[SP] : \min \quad & \sum_{i\in I}\sum_{j\in I} d_{ij}(v_{ij}) - \sum_{i\in I} \lambda_i x_i & \\
\text{s.t.} \quad & & \\
& \sum_{i\in I} m_i x_i \leq Q & \\
& v_{ij} \leq x_i & \forall i \in I, \forall j \in I \\
& v_{ij} \leq x_j & \forall i \in I, \forall j \in I \\
& v_{ij} \geq x_i + x_j - 1 & \forall i \in I, \forall j \in I \\
& x_i, v_{ij} \in \{0,1\} & \forall i, j \in I
\end{aligned}$$

The Lagrangian dual problem is:

$$\max_{\lambda,\mu\geq 0} LR(\lambda,\mu) = \max_{\lambda,\mu\geq 0}(\sum_{i\in I}\lambda_i - n_r\mu + n_b z_{SP}) \tag{3.10}$$

The Lagrangian master problem is:

$$\begin{aligned}
\max \quad & n_b\theta + \sum_{i\in I}\lambda_i - n_r\mu \\
\text{s.t.} \quad & \theta + \sum_{i\in I}\lambda_i x_i^h - \mu \leq \sum_{i\in I}\sum_{j\in I} d_{ij}v_{ij}^h \qquad\qquad h = 1,....H \\
& \lambda \text{ urs}, \mu \geq 0, \theta \leq 0.
\end{aligned}$$

Where H is the index set from the solution to [SP], and $\theta \leq 0$ is a result of the solution $y = 0$, $x = 0$.

3.3 Simulated Annealing

Since solving [OBP5] is time-consuming, we opt for a Simulated Annealing (SA) algorithm to be able to solve realistic instances. The SA algorithm starts with an initial solution that implements a strategy specifically designed to address the problem domain. The initial solution's fitness is evaluated by assessing its performance in relation to the objective function. The algorithm subsequently progresses through a sequence of iterations, wherein it generates a solution that is adjacent to the current one and assesses its level of suitability. The algorithm, in accordance with the acceptance criterion, determines whether to accept or reject the neighboring solution and subsequently updates the current solution accordingly.

3.3.1 Initial Solution and Neighbourhood Generation

One possible method for generating the initial solution is random allocation of items to batches. Despite its simplicity, this method may yield suboptimal solutions. The potential consequence of this situation is a decline in the effectiveness of the SA algorithm. Hence, we propose an initial solution based on k-means clustering with capacity allocation. The steps are:

1. Cluster all items using k-means, and the number of clusters equals the batch number.
2. Repeat the following steps until all items are assigned:
   - Select an item from one of the clusters
   - Check if the batch has capacity. If so, then assign the item to the selected batch. If not, move to the next batch.
   - Add the selected item to the solution with the chosen assignment.

Items are clustered according to their location within the warehouse. Nevertheless, the current clustering solution fails to take into account batch capacity. Therefore, we must assess the capacity of each cluster, and if it surpasses the limit, we proceed to transfer items to the subsequent batch that is in close proximity to the preceding cluster. The algorithm has a tendency to generate initial solutions that are reasonably good. The initial solutions are represented as a two-dimensional array with dimensions n_I by n_B.

The objective function from [OBP5] is used as the fitness function for SA. A neighbor solution is generated for each iteration in the search algorithm, and its fitness value is evaluated. A neighbor solution is generated by randomly selecting an item from the two dimensional array. The swap operation is executed on the chosen item. In order to identify the most effective swap, an evaluation is conducted on the capacity of each batch, and a decision is made regarding the batch swap within a loop iteration. In the event that the solution proposed by the neighbor surpasses the current solution in terms of quality, the search process will transition to the newly generated solution, resulting in an update to the best solution. When a suboptimal solution is generated, the search algorithm proceeds to evaluate the neighbor solution using an acceptance probability calculated as $\exp\left(\frac{current\_fitness - neighbor\_fitness}{temperature}\right)$. Subsequently, the temperature is modified through the process of multiplication with the cooling factor, thereby initiating a fresh iteration. The search process persists until the temperature reaches a value below the predetermined threshold, commonly referred to as the final temperature.

An iterative simulated annealing algorithm which is described in Algorithm 2. Initially, a set of 100 initial solutions is generated using the GenerateInitialSolutionWithKMeans() function. Subsequently, the best-performing solution is identified by evaluating its fitness using the EvaluateFitness() function. Next, the simulated annealing algorithm is initiated, and the resulting solution is obtained. Once a better solution is obtained using the SA algorithm, we proceed to reapply SA using the previously obtained solution. The loop is executed 10 times.

Algorithm 1 Simulated Annealing Algorithm (SA)

```
1: Initialization:
2:   Initialize: $current\_solution \leftarrow$ GenerateInitialSolutionWithKMeans()
3:       $current\_fitness \leftarrow$ EvaluateFitness($current\_solution$)
4:       $best\_solution \leftarrow current\_solution$
5:       $best\_fitness \leftarrow current\_fitness$
6:       $temperature \leftarrow initial\_temperature$
7:       $cooling\_factor \leftarrow cooling\_factor$
8: for $iteration \leftarrow 1$ to $num\_iterations$ do
9:   Generate a neighboring solution:
10:     $neighbor \leftarrow$ GenerateNeighbor($current\_solution$)
11:     $neighbor\_fitness \leftarrow$ EvaluateFitness($neighbor$)
12:   Cool down the temperature:
13:     $temperature \leftarrow temperature \times cooling\_factor$
14:   Calculate acceptance probability:
15:     $acceptance\_probability \leftarrow \exp\left(\frac{current\_fitness - neighbor\_fitness}{temperature}\right)$
16:   Accept or reject the neighbor solution:
17:   if $neighbor\_fitness \leq current\_fitness$ then
18:       $current\_solution \leftarrow neighbor$
19:       $current\_fitness \leftarrow neighbor\_fitness$
20:     if $neighbor\_fitness \leq best\_fitness$ then
21:         $best\_solution \leftarrow neighbor$
22:         $best\_fitness \leftarrow neighbor\_fitness$
23:     end if
24:   else if $acceptance\_probability \geq random(0,1)$ then
25:       $current\_solution \leftarrow neighbor$
26:       $current\_fitness \leftarrow neighbor\_fitness$
27:   end if
28:   if $temperature < final\_temperature$ then
29:     break
30:   end if
31: end for
32: return $best\_fitness,\ best\_solution$
33: Output: $best\_fitness,\ best\_solution$
```

Algorithm 2 Iterative Simulated Annealing Algorithm

```
1: Input: Instance, iteration, initial_temperature, cooling_factor
2: for $i \leftarrow 1$ to iteration do
3:   Find the best initial solution:
4:   if $i = 1$ then
5:       best_current ← [ ]
6:       best_fit ← [ ]
7:       num_iter ← 100
8:     for iter ← 1 to num_iter do
9:         current ← GenerateInitialSolutionWithKMeans()
10:        current_fitness ← EvaluateFitness(current)
11:        best_current.append(current)
12:        best_fit.append(current_fitness)
13:    end for
14:       min_ind ← index of minimum value in best_fit
15:       current ← best_curent[min_ind]
16:  else
17:    Use previous SA solution:
18:       current ← best_solution
19:  end if
20:    best_solution, best_fitness ← SA(initial_temperature, cooling_factor, current)
21: end for
22: return best_fitness, best_solution
23: Output: best_fitness, best_solution
```

Chapter 4

Numerical Testing

The proposed solution framework is tested on an open-source dataset from Kaggle called "Online Retail Dataset". All tests were performed using a 12th Gen Intel(R) Core(TM) i7-1255U 1.70 GHz computer with 16.0 GB of RAM. Python 3.9.12 was used for data analysis, and Gurobi 10.0.0 was used as the solver.

4.1 Data Set

The online retail dataset consists of transactional data of an online UK-based retailer from 01/12/2010 to 09/12/2011. The dataset has 8 columns with information about each transaction. The columns are described below:

1. InvoiceNo: Unique identifier for each transaction.
2. StockCode: Unique identifier for each item sold in the transaction.
3. Description: Description of the item sold.

4. Quantity: The quantity of the item sold in the transaction.

5. InvoiceDate: The date and time of the transaction.

6. UnitPrice: The price of one unit of the item sold.

7. CustomerID: Unique identifier for each customer.

8. Country: The country where the transaction was made.

The dataset consists of 541,909 transactions with a total of 3,719,908 items sold. The transactions correspond to customers from 38 different countries, with the majority of being from the United Kingdom. Figure 4.1 displays a heatmap for the number of total orders for each day.

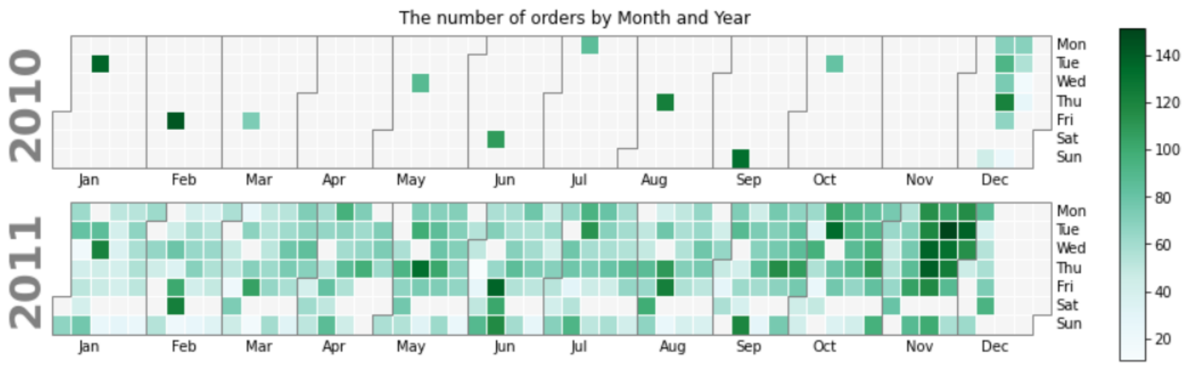

Figure 4.1: A heatmap of the number of orders by day, month, and year.

The number of items affects the batch and pick lists, as each robot carries a distinct capacity. The distribution of item quantity per order is shown in Figure 4.2. It is clear that certain orders display a quantity surpassing 1000 items. Due to the robot's carrying capacity limitations, allocating orders across multiple batches is necessary.

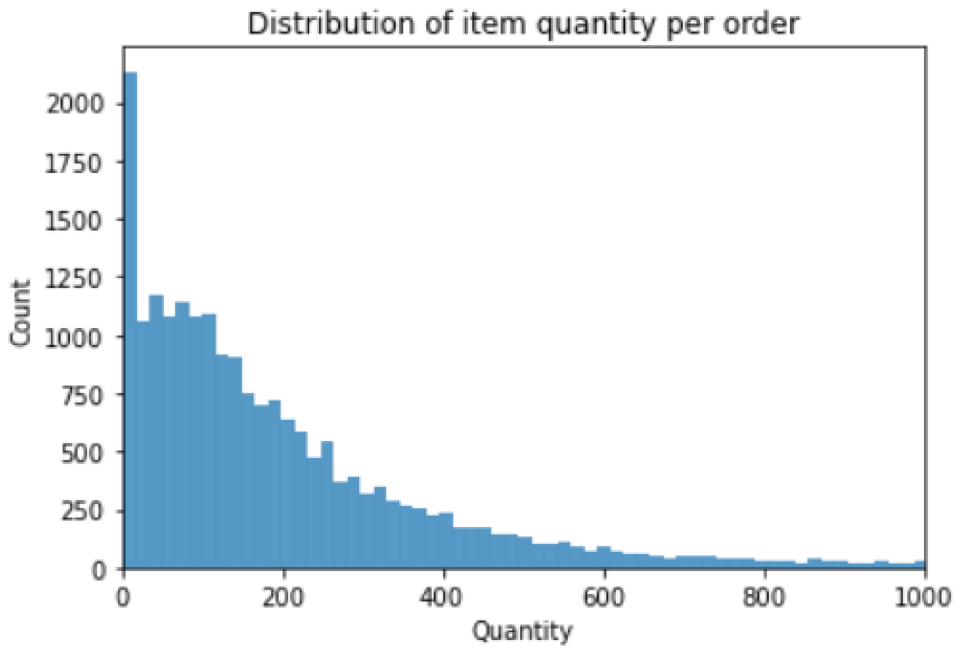

Figure 4.2: Distribution of item quantity per order.

4.2 Data Preprocessing

Although the online retail dataset contains 8 columns, relevant information is extracted from 3 columns. InvoiceNo is used to identify unique orders, StockCode for items, and Quantity for order capacity. 4,000 SKUs are used and located in an empty warehouse for this study. There are 20 aisles at the top and 20 aisles at the bottom. For each rack, the rectilinear distance is calculated from the assembly workstation up to the location of the item. Items are assigned to racks based on the item's frequency and distance. More frequent items are assigned to the shortest distance rack. As a result of this assignment, each item's aisle number, column, and row number are known. Table 4.1 shows a sample after preprocessing.

InvoiceNo	Description	Quantity	aisle	column	*row*
536365	CREAM CUPID HEARTS COAT HANGER	8	5	34.0	4.2
536365	KNITTED UNION FLAG HOT WATER BOTTLE	6	3	17.5	9.6
536365	RED WOOLLY HOTTIE WHITE HEART.	6	23	11.5	17.2
536365	SET 7 BABUSHKA NESTING BOXES	5	23	11.5	19.4
536365	WHITE HANGING HEART T-LIGHT HOLDER	6	1	10.0	3.0
536365	WHITE METAL LANTERN	6	4	28.0	8.6
536366	HAND WARMER RED POLKA DOT	6	35	83.5	25.2
536366	HAND WARMER UNION JACK	6	2	16.0	11.8
536367	ASSORTED COLOUR BIRD ORNAMENT	32	1	10.0	4.0
536367	BOX OF 6 ASSORTED COLOUR TEASPOONS	6	5	34.0	11.8
536367	BOX OF VINTAGE ALPHABET BLOCKS	2	24	17.5	20.6
536367	BOX OF VINTAGE JIGSAW BLOCKS	3	24	17.5	23.6

Table 4.1: A sample of the dataset after data preprocessing

The following parameters are calculated from Table 4.1.

$$a_i^k = \begin{cases} 1, & \text{if item } i \in I \text{ is on aisle } k \in K \\ 0, & \text{otherwise} \end{cases}$$

$$m_i = \text{the weight of item } i \in I$$

a_i^k are obtained from column 'aisle' and m_i from 'Quantity'.

4.3 Results And Numerical Analysis

Using the data, we solved all models using Gurobi 10.0.0 on instances ranging from 5 orders, 25 items, 5 batches to 25 orders, 241 items, and 40 batches. The results for each model are presented in the respective subsections below.

4.3.1 OBP1 Results

Table 4.2 presents the results for model [OBP1]. The columns display the number of order numbers (Orders), the number of Stock Keeping Units(Items), the total quantity of collected SKUs (Quantity), the number of batches (Batches), and the capacity of each instance. The optimal values are determined based on the total number of visited aisles, while the computational time is given in seconds. In order to provide a comprehensive comparison among all models, the total travel distance is computed by considering the route of each robot with respect to a determined optimal solution. We assume that the robot moves through the warehouse in one direction, starting from the item closest to the workstation and proceeding to the nearest uncollected item. The optimal solutions with routing for instances containing 5, 6, and 10 orders are displayed in Figure 4.3, 4.4, and 4.5, respectively. In the figures, the allocation of items to batches is visually represented by different colors. Robots at the workstation are allocated batches, which are symbolically represented by circles.

Orders	Items	Quantity	Batches	Capacity	Optimal value(# aisles)	Time(s)	Total travel distance(m)
5	25	173	5	100	24	0.08	1289.5
6	45	734	10	100	44	0.13	2079.5
7	46	814	10	100	45	0.11	2017
10	65	1026	15	100	64	0.21	2321
12	83	1383	20	200	82	0.26	2790.5
15	105	2120	25	400	104	0.37	3720.5
15	105	2120	20	400	infeasible	-	-
17	151	2724	20	400	infeasible	-	-
20	174	3319	25	400	infeasible	-	-
25	241	6908	40	600	infeasible	-	-

Table 4.2: Optimal values and computational time for [OBP1]

We can see that even though the solution is optimal for OBP1, the same robot is assigned to both the front and end aisles in the warehouse. This implies that one or more robots have to travel the warehouse from front to back twice, and their paths may cross, which may cause congestion.

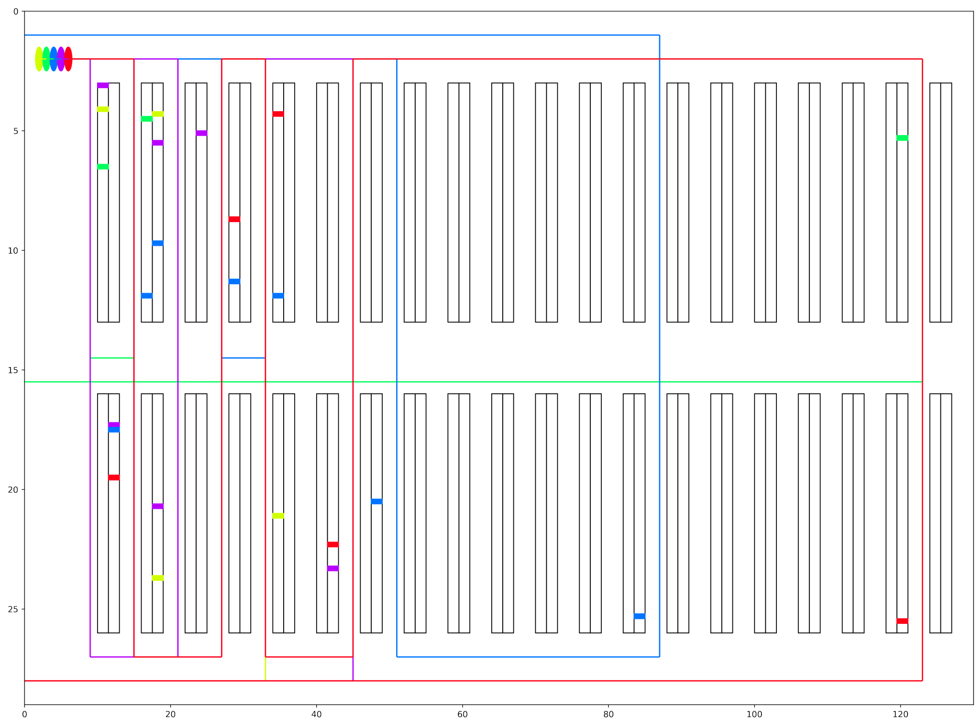

Figure 4.3: Display of the [OBP1] solution for 5 orders and 5 batches.

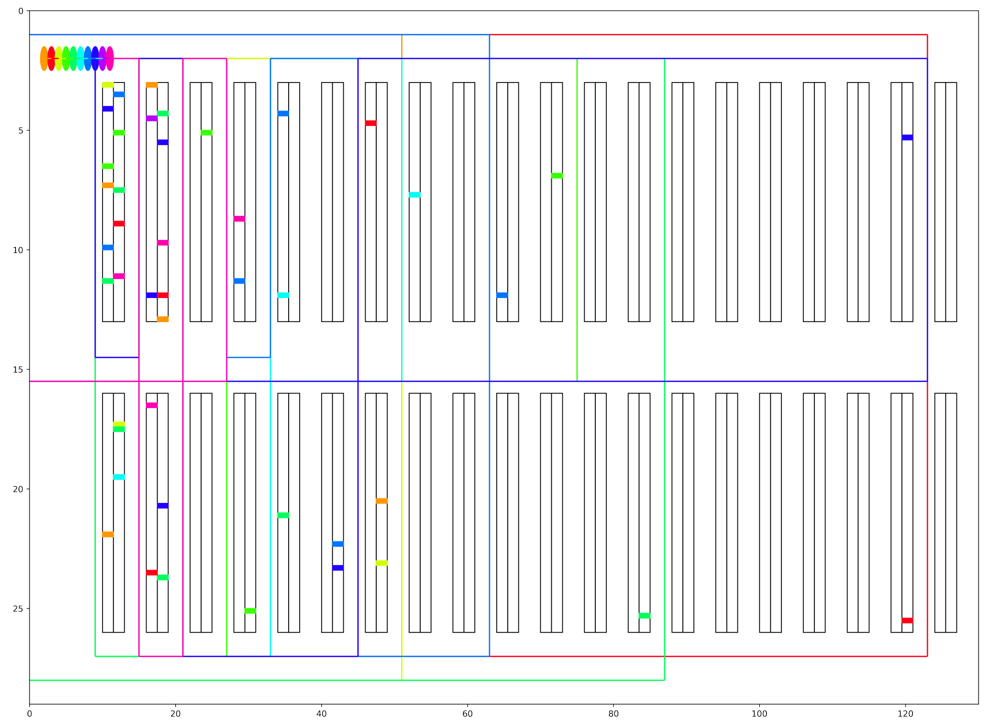

Figure 4.4: Display of the [OBP1] solution for 6 orders and 10 batches.

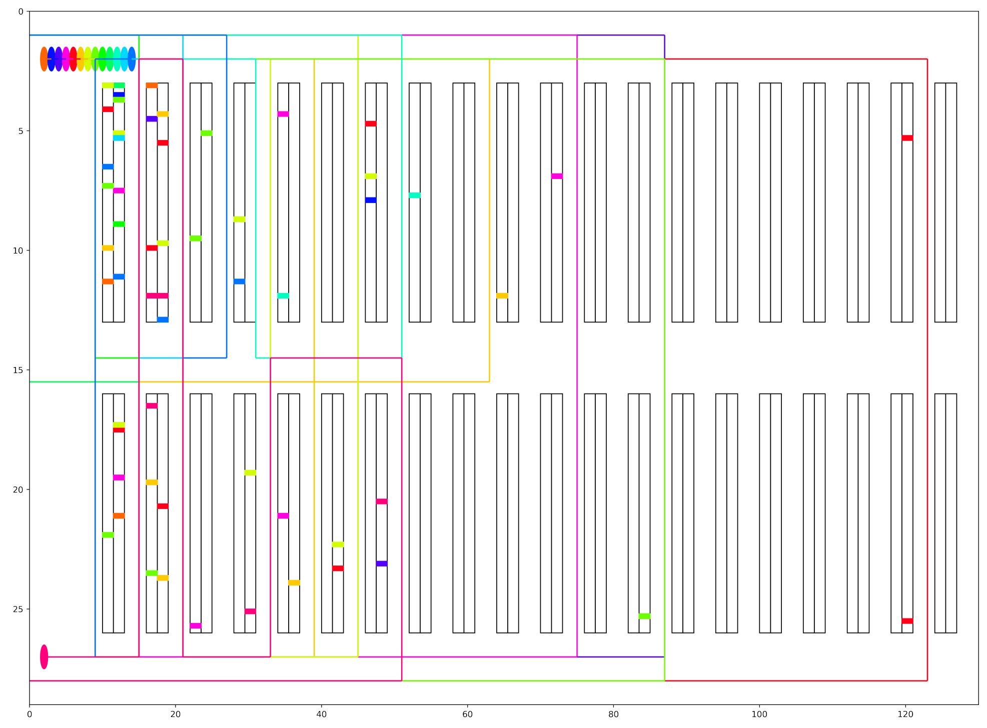

Figure 4.5: Display of the [OBP1] solution for 10 orders and 15 batches.

The outcomes of OBP1 show a significant occurrence of collisions, primarily attributed to the concurrent crossing paths of multiple robots.

4.3.2 OBP2 Results

Table 4.3 presents the solution results for [OBP2]. Optimal solutions that are illustrated in Figures 4.6, 4.7, and 4.8. Again, the solutions do not minimize distance and involve aisles far away from each other.

Orders	Items	Quantity	Batches	Capacity	Optimal value(# aisles)	Time(s)	Total travel distance(m)
5	25	173	5	100	5	0.09	1302.5
6	45	734	10	100	5	0.14	1981.5
7	46	814	10	100	5	0.17	1965.5
10	65	1026	15	100	5	0.22	2873
12	83	1383	20	200	5	0.32	3407
15	105	2120	25	400	5	0.45	4288.5
15	105	2120	20	400	infeasible	-	-
17	151	2724	20	400	infeasible	-	-
20	174	3319	25	400	infeasible	-	-
25	241	6908	40	600	infeasible	-	-

Table 4.3: Optimal values and computational time for [OBP2]

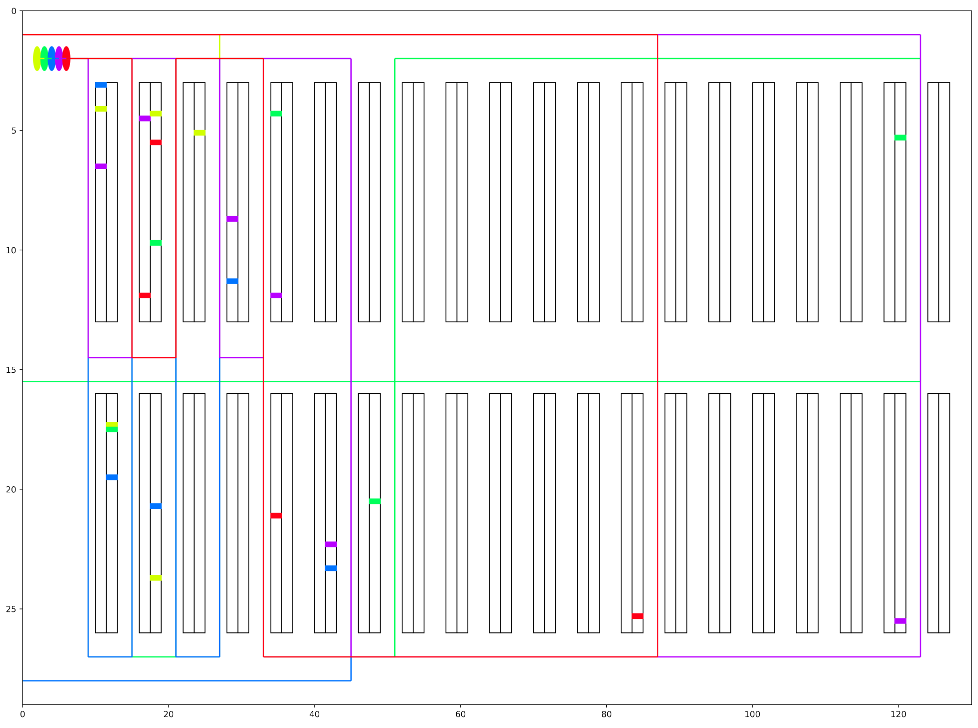

Figure 4.6: Display of the [OBP2] solution for 5 orders and 5 batches.

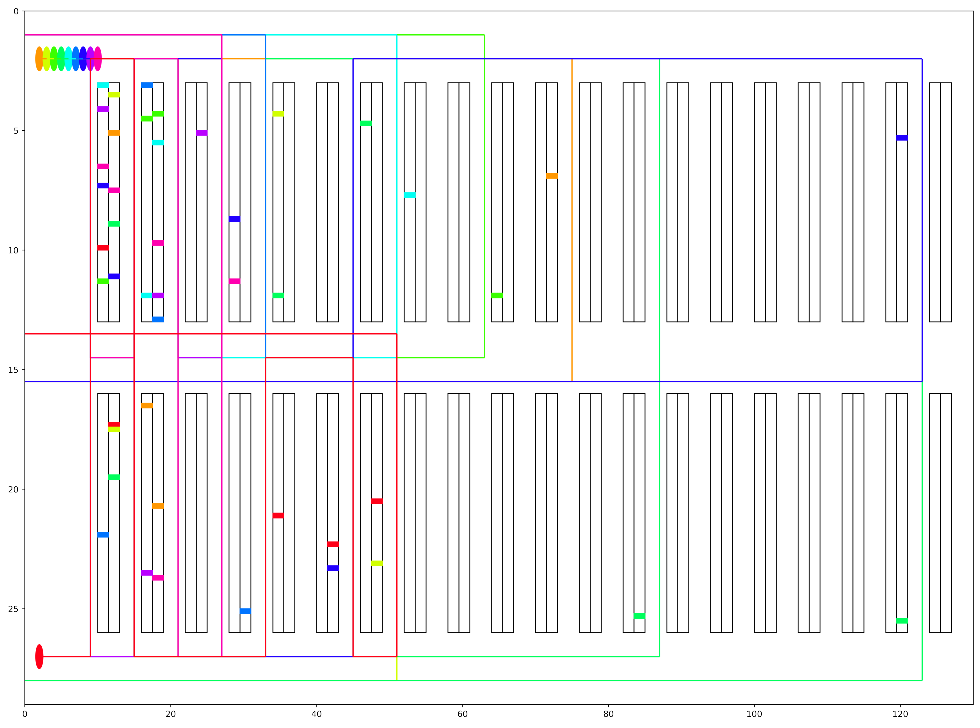

Figure 4.7: Display of the [OBP2] solution for 6 orders and 10 batches.

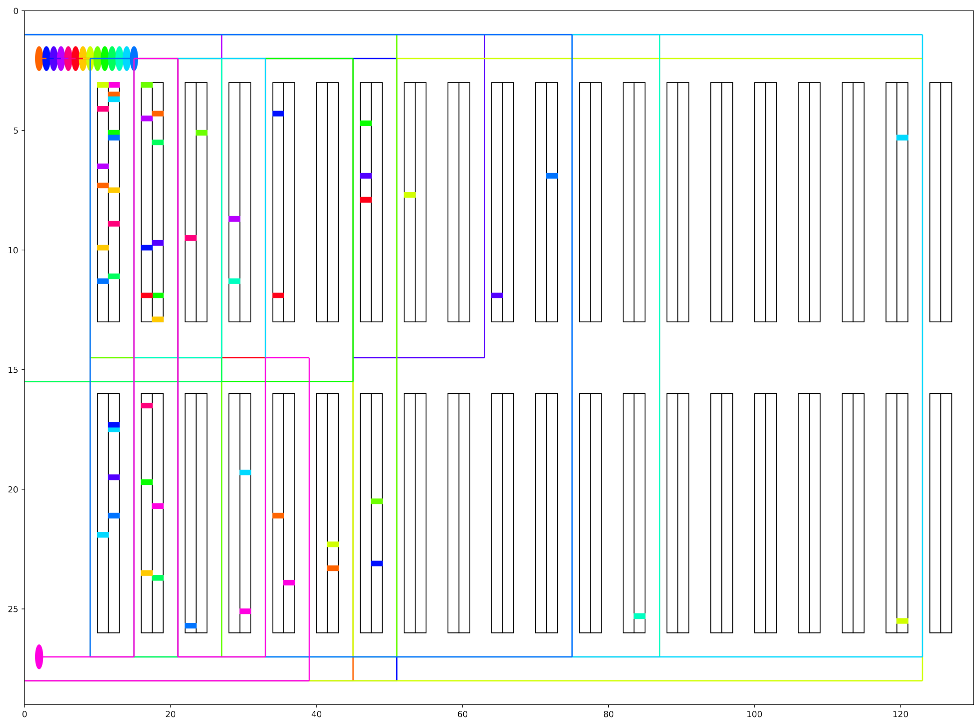

Figure 4.8: Display of the [OBP2] solution for 10 orders and 15 batches.

Solutions from [OBP2] may lead to congestion primarily attributed to the concurrent crossing paths of multiple robots. Additionally, the cumulative travel distance of all robots is large.

4.3.3 OBP3 Results

Table 4.4 presents the implementation results for various order numbers using the same representation as Table 4.2. The optimal solutions for all instances listed in Table 4.4 are depicted in Figures 4.9, 4.10, and 4.11. The parameters used for OBP3 are as follows: the length parameter (L) is set to 130, the width parameter (W) is set to 29, and the value of the parameter K is 42.

Orders	Items	Quantity	Batches	Capacity	Optimal value(m)	Time(s)	Total travel distance(m)
5	25	173	5	100	77.6	1.45	1038.5
6	45	734	10	100	81.6	22.99	1943.5
7	46	814	10	100	92.9	35.10	1786.5
10	65	1026	15	100	92.9	71.59	2418
12	83	1383	20	200	92.9	239.12	3012
15	105	2120	25	400	148.2*	900.00	4243
15	105	2120	20	400	infeasible	-	-
17	151	2724	20	400	infeasible	-	-
20	174	3319	25	400	infeasible	-	-
25	241	6908	40	600	infeasible	-	-

* Best found feasible solution

Table 4.4: Optimal values and computational time for [OBP3]

Figure 4.9, 4.10, and 4.11 show that [OBP3] is good at clustering items for batches. However, this model cannot solve more than 10 orders.

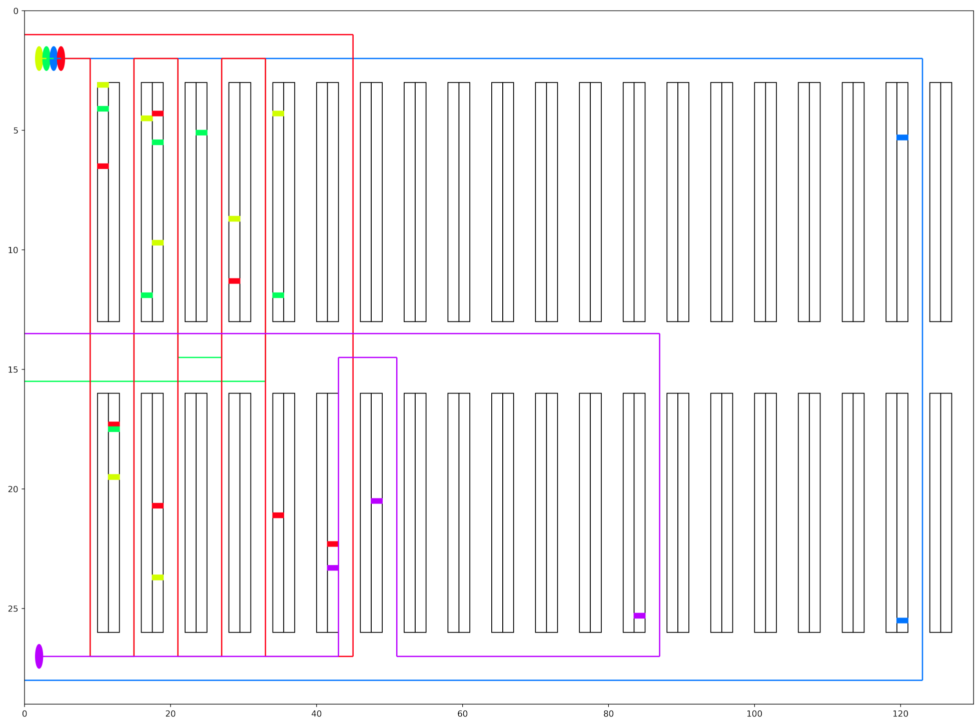

Figure 4.9: Display of the [OBP3] solution for 5 orders and 5 batches.

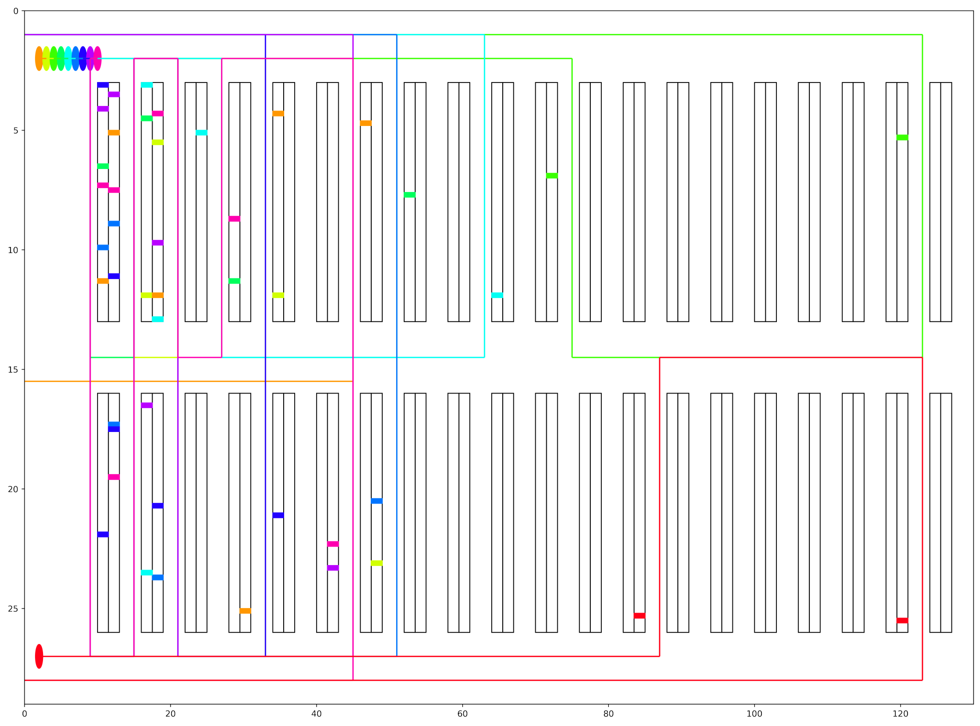

Figure 4.10: Display of the [OBP3] solution for 6 orders and 10 batches.

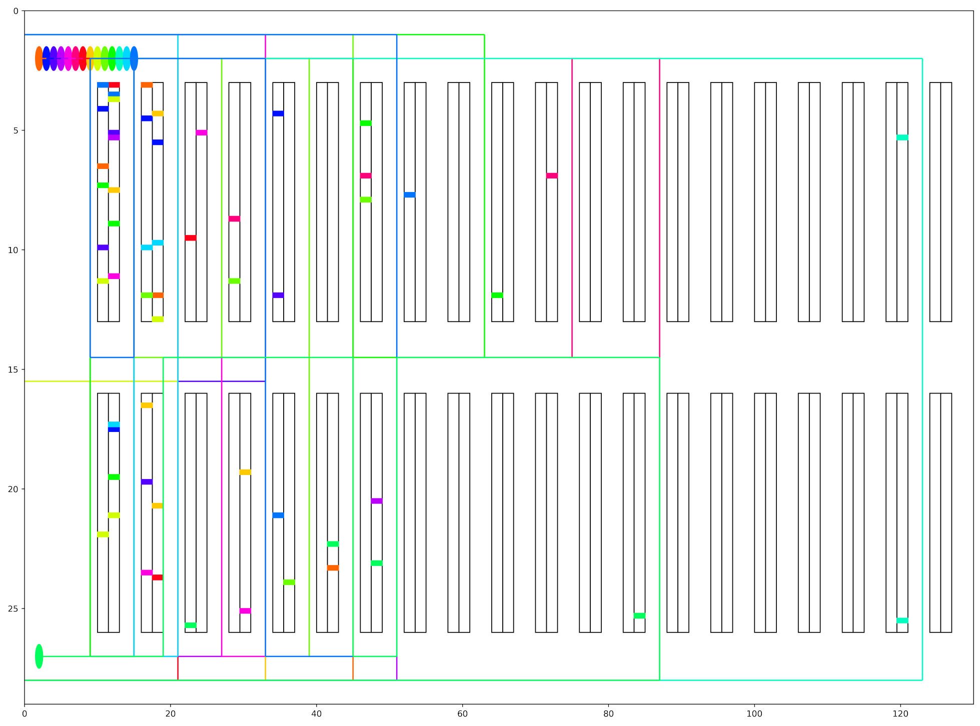

Figure 4.11: Display of the [OBP3] solution for 10 orders and 15 batches.

The results of [OBP3] indicate a slightly better overall travel distance compared to that of [OBP1] and [OBP2]. Nevertheless, the issue of intersecting paths and the potential for collisions remains a significant concern.

4.3.4 OBP4 Results

Table 4.5 presents the results for various order numbers using the same representation as Table 4.2. The optimal solutions visualized in Figure 4.12, 4.13 and 4.14.

Orders	Items	Quantity	Batches	Capacity	Optimal value(m)	Time(s)	Total travel distance(m)
5	25	173	5	100	32.3	0.98	890.5
6	45	734	10	100	33.7	22.56	1386.5
7	46	814	10	100	44.0	19.91	1508.5
10	65	1026	15	100	20.2	486.16	1666.5
12	83	1383	20	200	11.9	376.71	1972.0
15	105	2120	25	400	18.3*	900	2547
15	105	2120	20	400	51.5*	900	2677.5
17	151	2724	20	400	108.6*	900	3529.5
20	174	3319	25	400	90.6*	900	4434
25	241	6908	40	600	120.3*	900	6329.5

* Best found feasible solution

Table 4.5: Optimal values and computational time for [OBP4]

Based on Table 4.5, it is evident that the computational times grow exponentially and become intensive for 12 orders or more. Therefore, a time limit of 900 seconds is imposed, and the best found feasible solutions are provided.

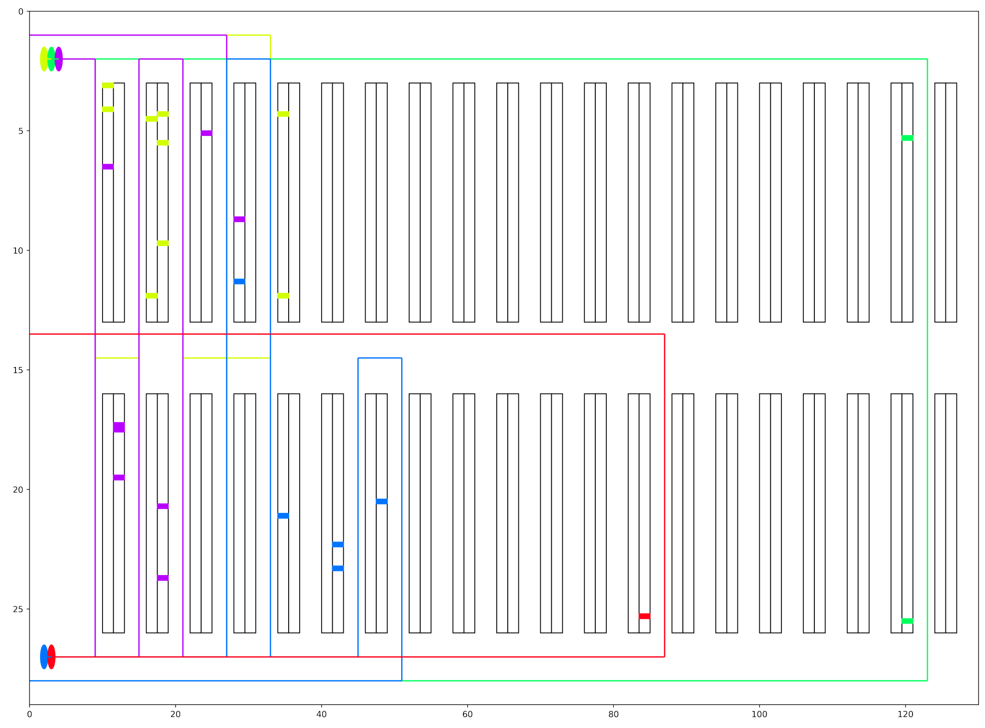

Figure 4.12: Display of the [OBP4] solution for 5 orders and 5 batches.

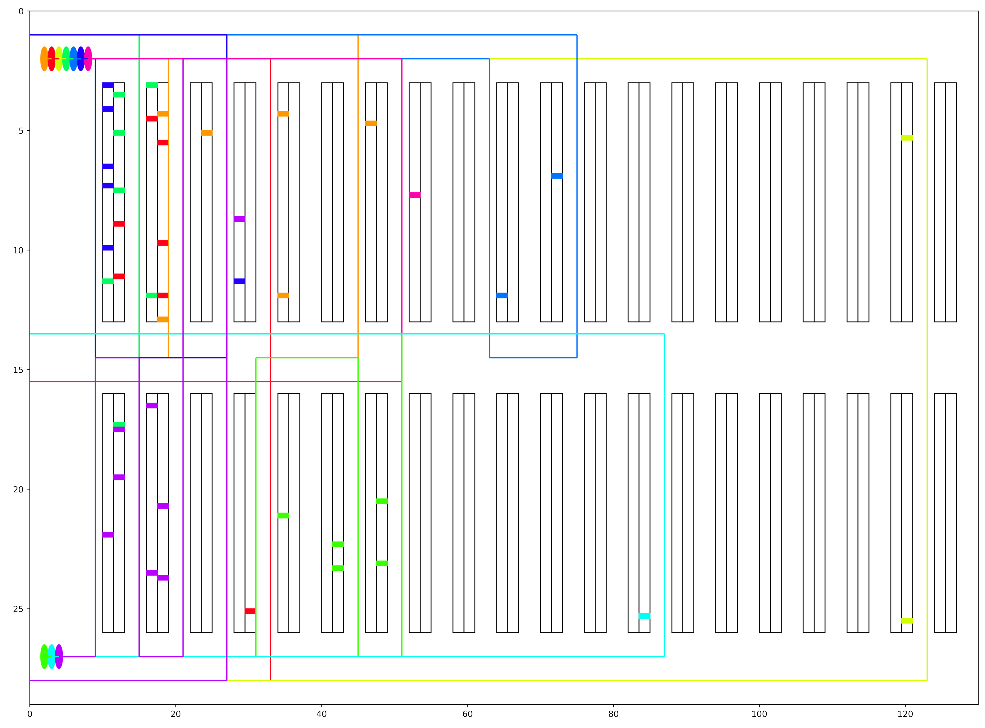

Figure 4.13: Display of the [OBP4] solution for 6 orders and 10 batches.

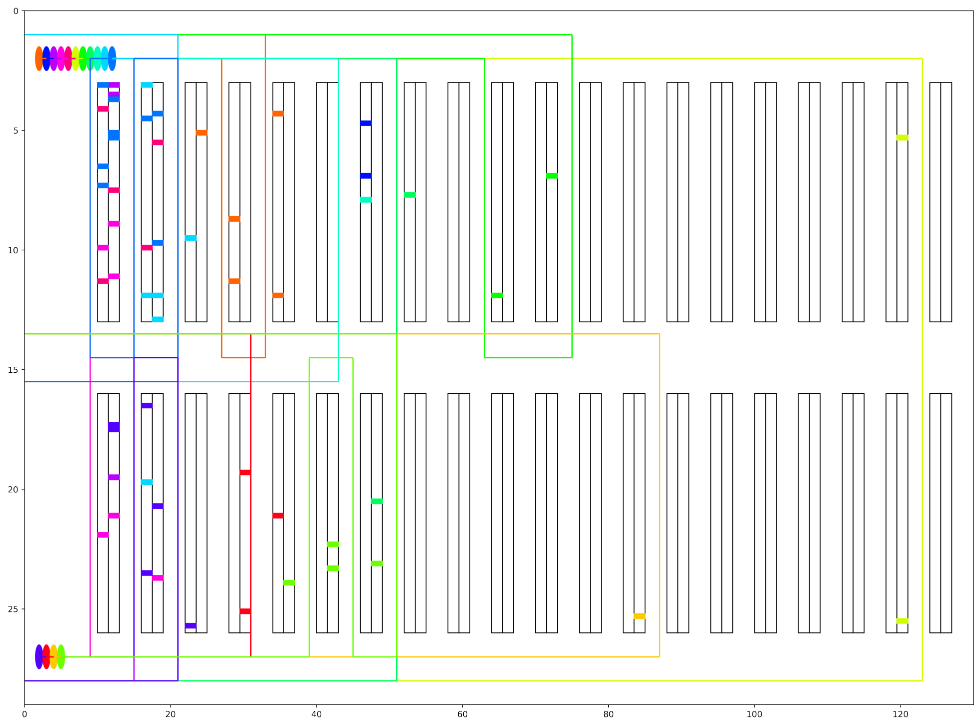

Figure 4.14: Display of the [OBP4] solution for 10 orders and 15 batches.

The effectiveness of [OBP4] in minimizing the distance traveled by each robot is apparent. Furthermore, it is important to acknowledge that [OBP4] has fewer decision variables than [OBP3]. However, there is still room for improvement in terms of finding more efficient travel paths.

4.3.5 OBP5 Results

Table 4.6 presents the solution results for [OBP5] using the same representation as Table 4.2. The solutions are displayed in Figures 4.15,4.16, and 4.17. According to Table 4.6, the solution time is very large for instances as small as 6 orders and 25 items. We set a time limit of 900 seconds and took the best feasible solution.

Orders	Items	Quantity	Batches	Capacity	Optimal value(m)	Time(s)	Total travel distance(m)
5	25	173	5	100	1365.40	24.42	659
6	45	734	10	100	1737*	900	1170.5
7	46	814	10	100	2059.9*	900	1159
10	65	1026	15	100	1968.8*	900	1548.5
12	83	1383	20	200	1151.6*	900	1646
15	105	2120	25	400	1018.8*	900	1835
15	105	2120	20	400	1877.4*	900	1611.5
17	151	2724	20	400	5449.5*	900	1845.5
20	174	3319	25	400	4840.9*	900	2306
25	241	6908	40	600	3668.1*	900	3237

* Best found feasible solution

Table 4.6: Optimal values and computational time for [OBP5]

Figure 4.15, 4.16, and 4.17 show that [OBP5] is good at clustering items for batches and minimizing total travel distance for each robot.

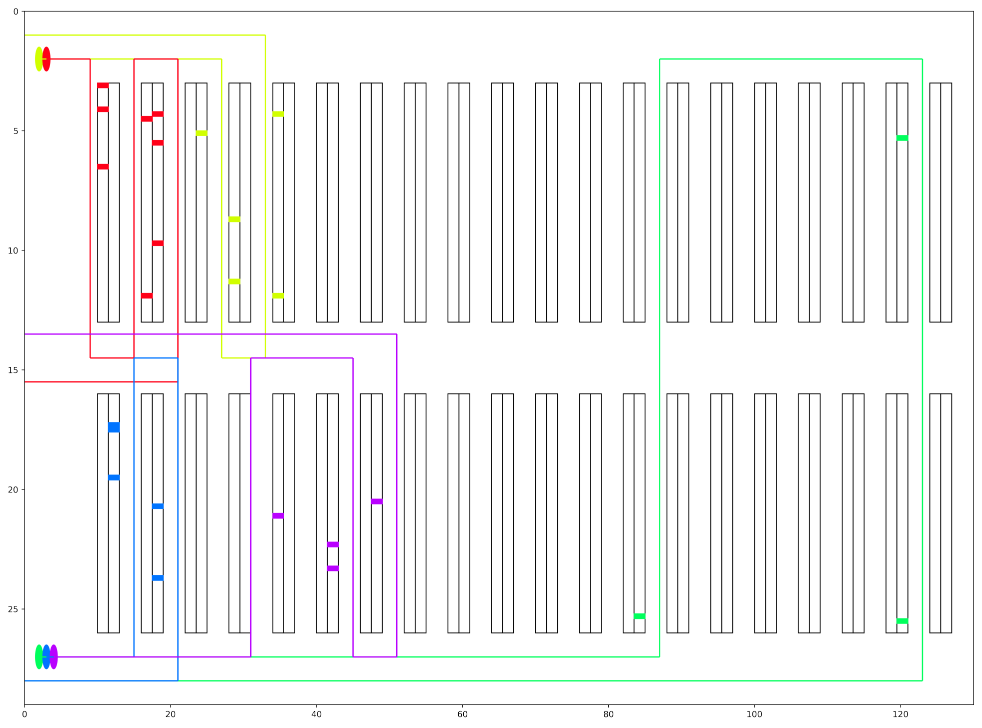

Figure 4.15: Display of the [OBP5] solution for 5 orders and 5 batches.

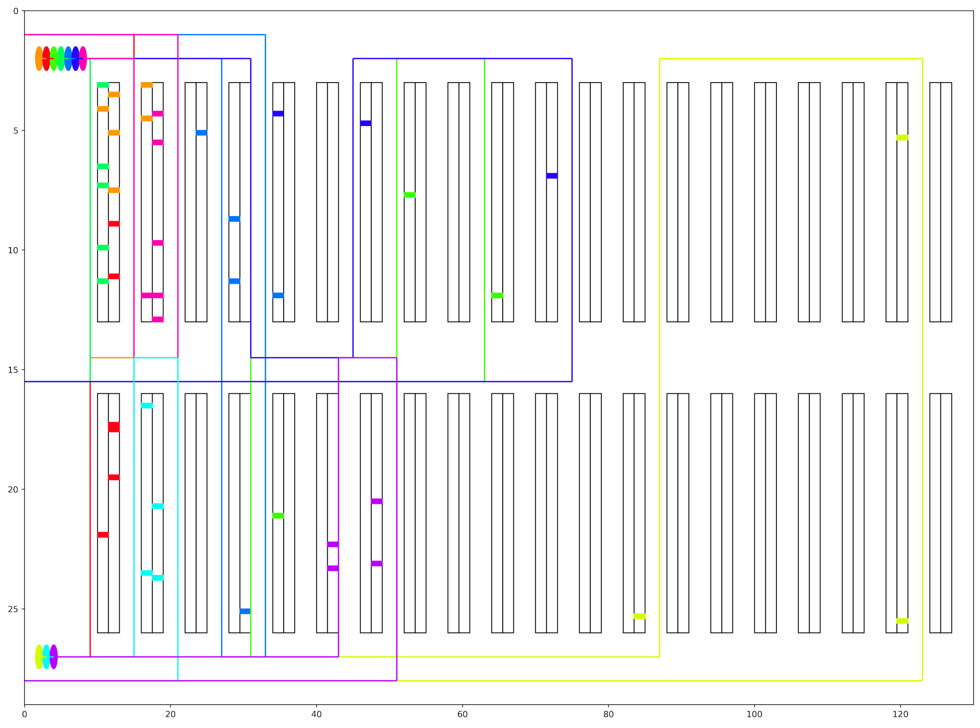

Figure 4.16: Display of the [OBP5] solution for 6 orders and 10 batches.

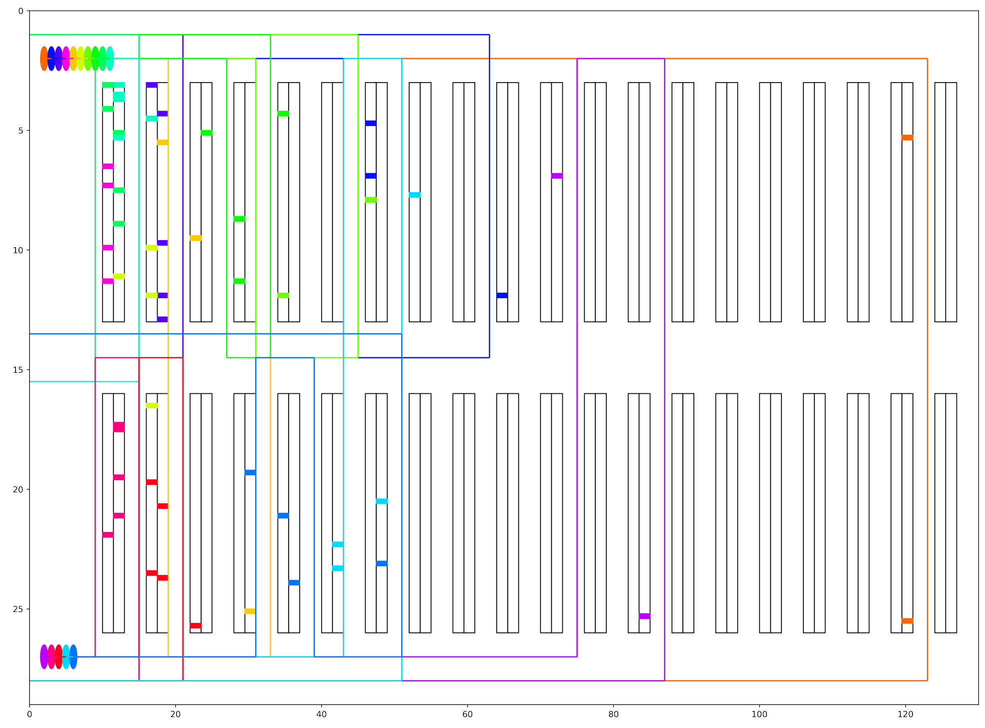

Figure 4.17: Display of the [OBP5] solution for 10 orders and 15 batches.

Figure 4.18 displays the travel distance of 5 robots for all models with 5 orders and 5 batches. Figure 4.19 displays the total travel distance of all robots for all models ranging from 5 orders to 25 orders. According to Figure 4.18 and 4.19 [OBP5] achieves the lowest travel distance among all the robots.

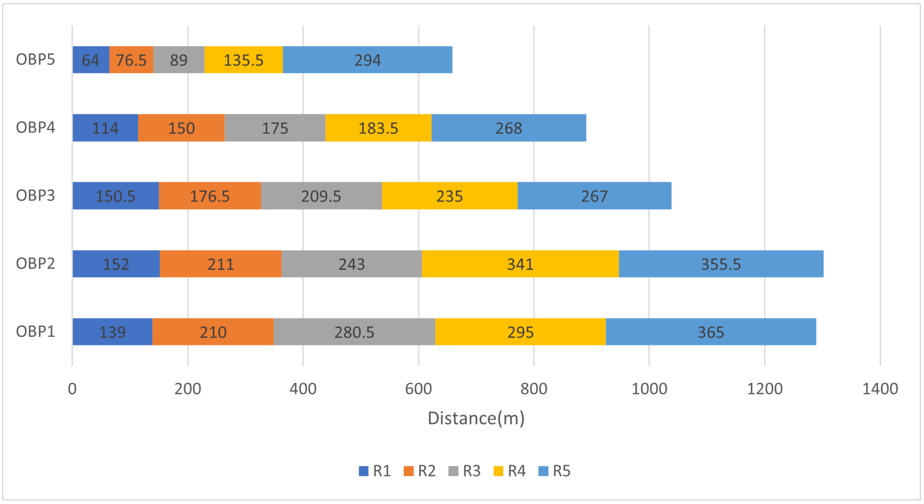

Figure 4.18: Robot's travel distance comparison for all models

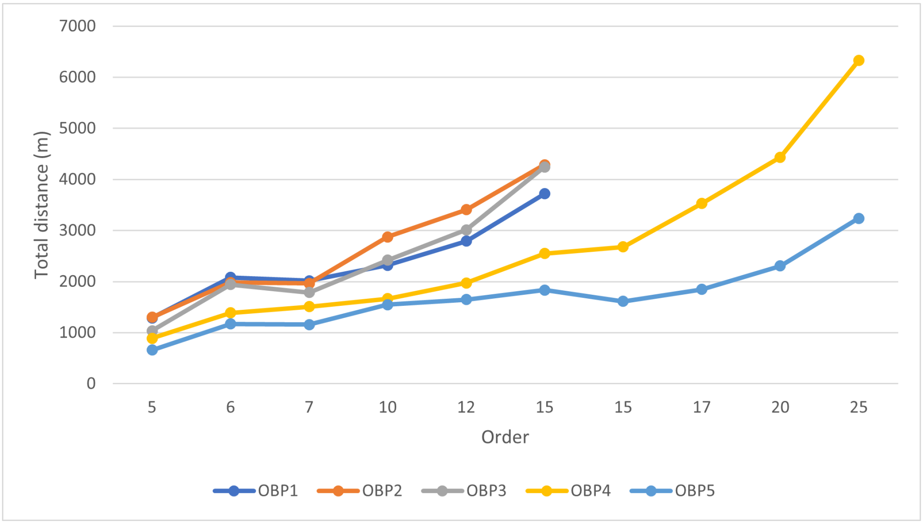

Figure 4.19: Total travel distance comparison for all models

4.3.6 Iterative SA and Lagrangian Relaxation Results

In this section, final_temperature, cooling_factor, and num_iterations for the simulated annealing algorithm, are set to 0.0001, 0.99, and 1000, respectively.

Table 4.7 presents results for various instances using the representation as Table 4.2. To evaluate the quality of the SA solution, the second last column (Improvement) gives the improvement over the OBP5 objective calculated as $\left(100 * \frac{\text{SA}-\text{OBP5}}{\text{OBP5}}\right)$. The last column (GAP) gives the SA solution quality with respect to the Lagrangian lower bound, calculated as $\left(100 * \frac{\text{LR}-\text{SA}}{\text{SA}}\right)$.

Lagrangian Relaxation provides a good quality lower bound, but it is not particularly useful in terms of computing time. SA, on the other hand, provides good quality solutions as assessed by the Lagrangian lower bound and the direct solutions of [OBP5]. Additionally, Figure 4.20 and 4.22 show that SA is able to find solutions that effectively group items and reduce robot travel distance and congestion.

					OBP5				SA			LR			
Orders	Items	Quantity	Batches	Capacity	Objective	Gap	LP Bound	Time(s)	Objective	Time(s/it)	Temp	Objective	Time(s)	Improvement	GAP
5	25	173	5	100	1365.40	14.80	1162.7	24.42	1365.39	1.55	100	1365.399	324.5	0.00	0.00
6	45	734	10	100	1737	84.00	278.72	900	1604.1	6.27	100	1597.42	393.96	7.65	0.42
7	46	814	10	100	2059.9	82.60	357.51	900	1965.4	6.46	100	1781.06	407.15	4.59	10.35
10	65	1026	15	100	1968.8	100	0	900	1931.1	16.5	100	1665.67	1781.76	1.91	15.94
12	83	1383	20	200	1151.6	100	0	900	1077.4	33.09	10	1054	6178.31	6.44	2.22
15	105	2120	20	400	1877.4	100	0	900	1974.1	52.61	10	1822.8	23966.99	-5.15	8.30
15	105	2120	25	400	1018.8	100	0	900	1042	62.56	100			-2.28	-
17	151	2724	20	400	5449.5	100	0	900	5504.3	107.27	100	-	-	-1.01	-
20	174	3319	25	400	4840.9	100	0	900	4820.2	181.26	100	-	-	0.43	-
25	241	6908	40	600	3668.1	100	0	900	3684.8	501.56	100	-	-	-0.46	-
												Average		1.21	6.20

Table 4.7: Comparision of SA and [OBP5] with respect to the LR lower bounds.

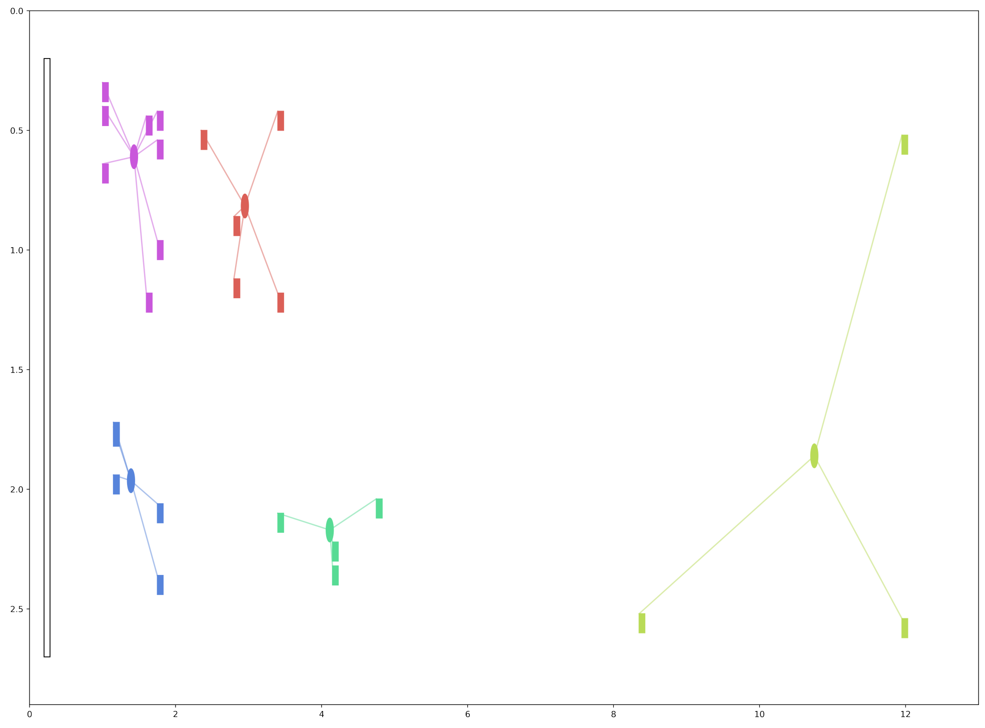

Figure 4.20: Assigned items' distance from the center of batches for SA solution for 5 orders and 5 batches.

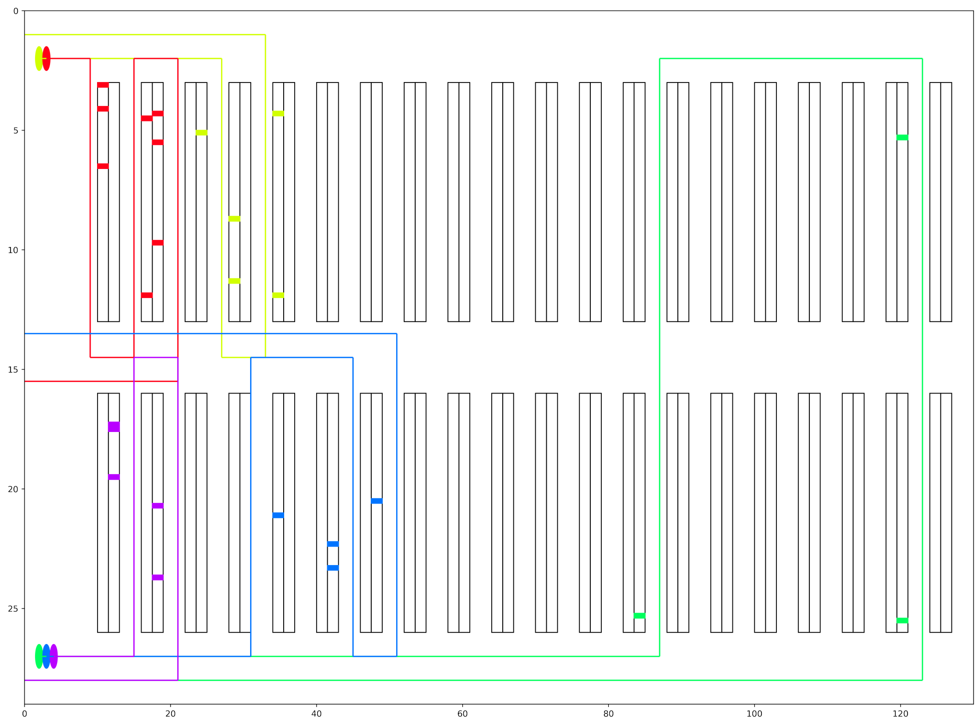

Figure 4.21: Display of the SA solution for 5 orders and 5 batches.

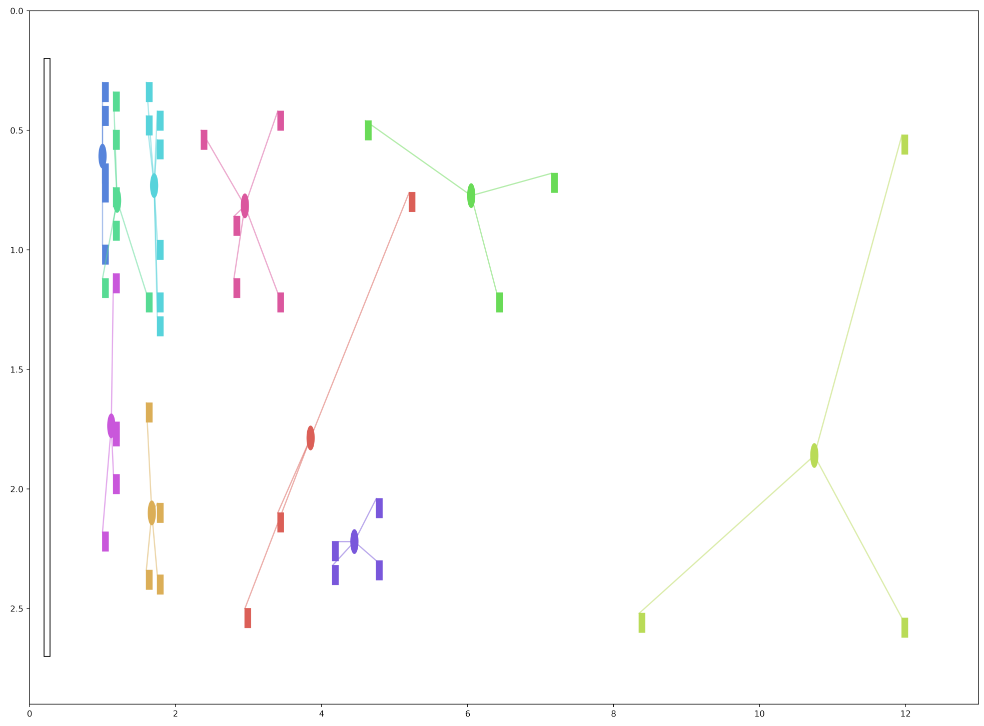

Figure 4.22: Assigned items' distance from the center of batches for SA solution for 6 orders and 10 batches.

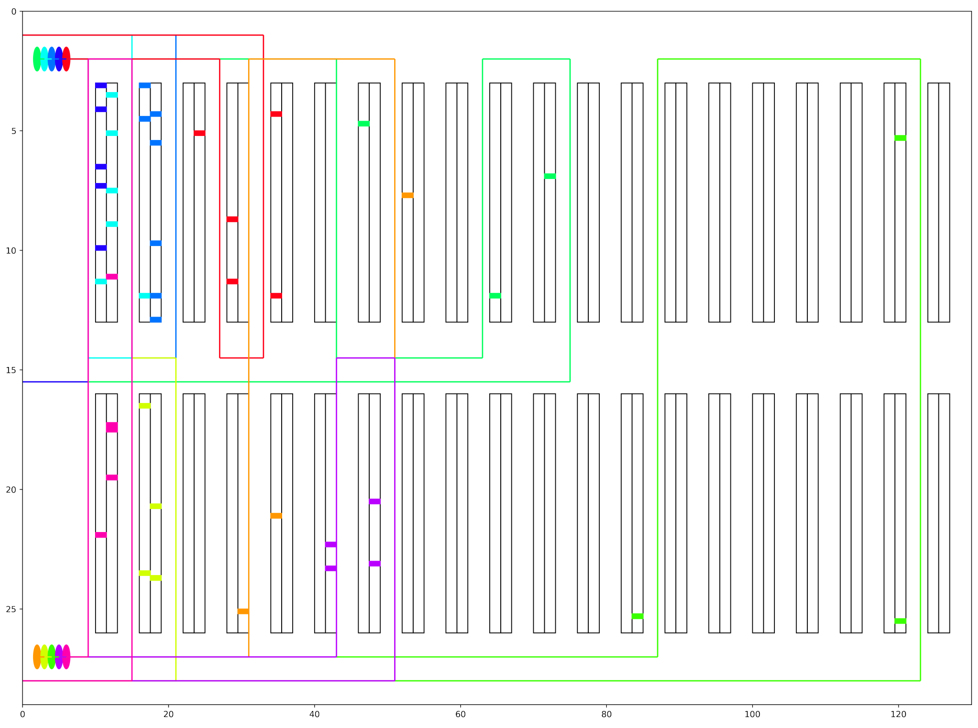

Figure 4.23: Display of the SA solution for 6 orders and 10 batches.

The iterative SA algorithm achieves comparable solutions to OBP5 in short computational times. It finds better solutions for smaller instances but falls slightly short for the large instances. With respect to LR, Sa finds solutions within an average of 6.2% but ranges from 0% gap to 15.94%.

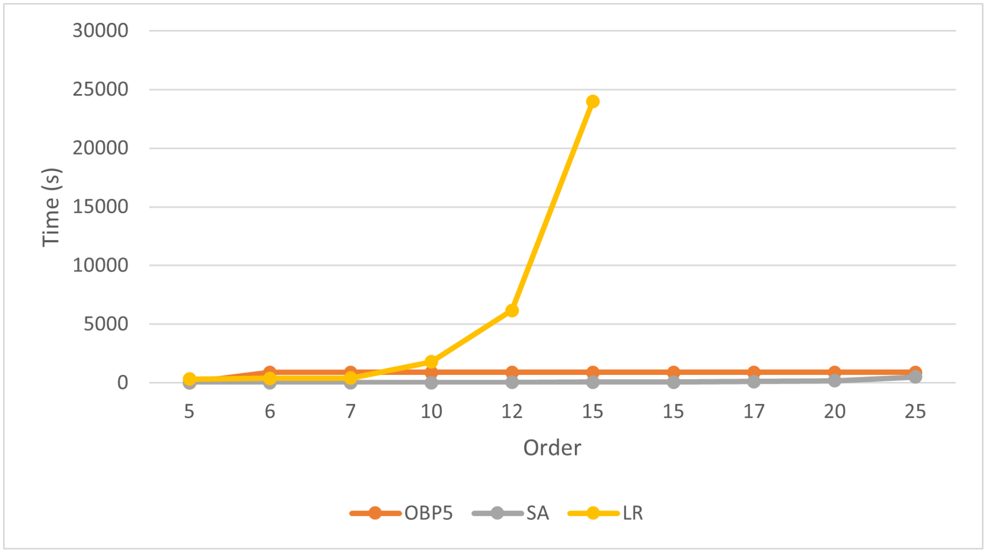

Figure 4.24: Time comparison of [OBP5], SA and Lagrangian Relaxation

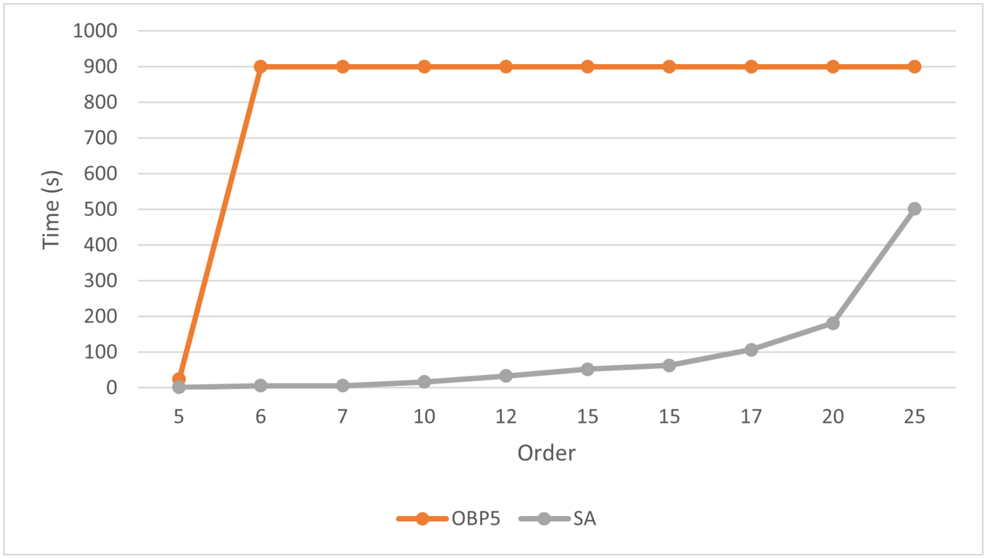

Figure 4.25: Time comparison of [OBP5] and SA

When comparing the iterative SA algorithm to the direct solution of [OBP5] in terms of time, it is shown in Figure 4.24 and 4.25 that the iterative SA approach exhibited superior performance.

Chapter 5

Conclusion

The use of automated warehouse systems has grown considerably during and after the COVID-19 pandemic. This shift is driven by the need to enhance efficiency, minimize human contact, and expedite delivery. The order-picking process, which is an essential element of warehouse operations, has a substantial influence on both customer satisfaction and operational expenses.

In this work, we presented, modeled, and provided solution methods for the order batching problem in automated warehouses. We provided different mathematical models and compared them based on clustering and distance minimization capabilities. We found that the model based on the generalized quadratic assignment problem performs the best. It suffers, however, from being computationally burdensome. To remedy this, we presented a Lagrangian relaxation and a Simulated Annealing metaheuristic.

Numerical testing revealed that the proposed simulated algorithm provides a high-quality solution as assessed by the Lagrangian lower bound and in comparison to the direct solution of the mathematical model. It, therefore, presents a powerful tool to solve realistic instances and can be directly integrated into warehouse management systems.

Future research can be devoted to increasing the efficiency of the proposed Lagrangian relaxation approach as it potentially provides a good quality bound. Alternatively, other relaxations are worth exploring. In addition, different features of the order batching problem, such as congestion, can be modeled explicitly. Another possible direction is to integrate different problems in warehouse management, such as layout optimization and order batching.

References

Lin Xie, Hanyi Li, and Laurin Luttmann. Formulating and solving integrated order batching and routing in multi-depot AGV-assisted mixed-shelves warehouses. *European Journal of Operational Research*, 307, 09 2022. doi: 10.1016/j.ejor.2022.08.047.

Claire Salles. Immediacy and its hidden infrastructure: When amazon extends its delivery times during the covid-19 pandemic. *img journal*, 2(3):380–395, Jan. 2020. doi: 10.6092/issn.2724-2463/12265. URL `https://img-journal.unibo.it/article/view/12265`.

Charles Petersen and Roger Schmenner. An evaluation of routing and volume-based storage policies in an order picking operation. *Decision Sciences*, 30:481 – 501, 06 2007. doi: 10.1111/j.1540-5915.1999.tb01619.x.

Nils Boysen, René De Koster, and Felix Weidinger. Warehousing in the e-commerce era: A survey. *European Journal of Operational Research*, 277, 08 2018. doi: 10.1016/j.ejor.2018.08.023.

Edward Frazelle. *Warehouse Operations*. Supply Chain Strategy: The Logistics of Supply Chain Management. McGraw-Hill Education, New York, first edition. edition, 2002. ISBN 9780071375993. URL `https://www.accessengineeringlibrary.com/content/book/9780071375993/chapter/chapter8`.

Kaveh Azadeh, René De Koster, and Debjit Roy. Robotized and Automated Warehouse Systems: Review and Recent Developments. *Transportation Science*, 53:917–945, 06 2019. doi: 10.1287/trsc.2018.0873.

Qing Guo Zheng Zhang, Juan Chen. Application of automated guided vehicles in smart automated warehouse systems: A survey. *Computer Modeling in Engineering & Sciences*, 134(3):1529–1563, 2023. ISSN 1526-1506. doi: 10.32604/cmes.2022.021451. URL http://www.techscience.com/CMES/v134n3/49756.

Kees Jan Roodbergen and Iris Vis. A survey of literature on automated storage and retrieval systems. *European Journal of Operational Research*, 194:343–362, 04 2009. doi: 10.1016/j.ejor.2008.01.038.

Warren Hausman, Leroy Schwarz, and Stephen Graves. Optimal storage assignment in automatic warehousing systems. *Management Science*, 22:629–638, 02 1976. doi: 10.1287/mnsc.22.6.629.

Debjit Roy, Shobhit Nigam, René de Koster, Ivo Adan, and Jacques Resing. Robot-storage zone assignment strategies in mobile fulfillment systems. *Transportation Research Part E: Logistics and Transportation Review*, 122:119–142, 2019. ISSN 1366-5545. doi: https://doi.org/10.1016/j.tre.2018.11.005. URL https://www.sciencedirect.com/science/article/pii/S1366554518304307.

Hyun-Jung Kim, Cristobal Pais, and Zuo-Jun Max Shen. Item assignment problem in a robotic mobile fulfillment system. *IEEE Transactions on Automation Science and Engineering*, 17(4):1854–1867, 2020. doi: 10.1109/TASE.2020.2979897.

Xiaowei Li, Guowei Hua, Anqiang Huang, Jiuh-Biing Sheu, T.C.E. Cheng, and Fengquan Huang. Storage assignment policy with awareness of energy consumption in the kiva mobile fulfilment system. *Transportation Research Part E: Logistics and Transportation Review*, 144:102158, 2020a. ISSN 1366-5545. doi: https://doi.org/10.1016/j.tre.2020.102158. URL https://www.sciencedirect.com/science/article/pii/S1366554520308036.

T. Lamballais, Debjit Roy, and René De Koster. Estimating performance in a robotic mobile fulfillment system. *European Journal of Operational Research*, 256, 07 2016. doi: 10.1016/j.ejor.2016.06.063.

K. L. Keung, C. K. M. Lee, P. Ji, and Kam K. H. Ng. Cloud-based cyber-physical robotic mobile fulfillment systems: A case study of collision avoidance. *IEEE Access*, 8:89318–89336, 2020. doi: 10.1109/ACCESS.2020.2992475.

K. L. Keung, C. K. M. Lee, and P. Ji. Mobile robots charging assignment problem with time windows in robotic mobile fulfilment system. In *2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, pages 1329–1333, 2019. doi: 10.1109/IEEM44572.2019.8978958.

C.K.M. Lee, Bingbing Lin, K.K.H. Ng, Yaqiong Lv, and W.C. Tai. Smart robotic mobile fulfillment system with dynamic conflict-free strategies considering cyber-physical integration. *Advanced Engineering Informatics*, 42:100998, 2019. ISSN 1474-0346. doi: https://doi.org/10.1016/j.aei.2019.100998. URL https://www.sciencedirect.com/science/article/pii/S1474034619305713.

K.L. Keung, C.K.M. Lee, and P. Ji. Data-driven order correlation pattern and storage location assignment in robotic mobile fulfillment and process automation system. *Advanced Engineering Informatics*, 50:101369, 2021. ISSN 1474-0346. doi:

https://doi.org/10.1016/j.aei.2021.101369. URL https://www.sciencedirect.com/science/article/pii/S1474034621001221.

Zhe Yuan and Yeming Gong. Bot-in-time delivery for robotic mobile fulfillment systems. *IEEE Transactions on Engineering Management*, PP:1–11, 01 2017. doi: 10.1109/TEM.2016.2634540.

Bipan Zou, Yeming Gong, Xianhao Xu, and Zhe Yuan. Assignment rules in robotic mobile fulfilment systems for online retailers. *International Journal of Production Research*, pages 1–18, 05 2017. doi: 10.1080/00207543.2017.1331050.

Shasha Wu, Cheng Chi, Wei Wang, and Yaohua Wu. Research of the layout optimization in robotic mobile fulfillment systems. *International Journal of Advanced Robotic Systems*, 17:172988142097854, 11 2020. doi: 10.1177/1729881420978543.

John Enright and Peter Wurman. Optimization and coordinated autonomy in mobile fulfillment systems. *AAAI Workshop - Technical Report*, 01 2011.

Peter Wurman, Raffaello D'Andrea, and Mick Mountz. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI Mag.*, 29:1752–, 01 2007.

Xiying Yang, Guowei Hua, Linyuan Hu, T.C.E. Cheng, and Anqiang Huang. Joint optimization of order sequencing and rack scheduling in the robotic mobile fulfilment system. *Computers Operations Research*, 135:105467, 2021. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2021.105467. URL https://www.sciencedirect.com/science/article/pii/S0305054821002185.

Nils Boysen, Dirk Briskorn, and Simon Emde. Parts-to-picker based order processing in a rack-moving mobile robots environment. *European Journal of Operational Research*, 262(2):550–562, 2017. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2017.03.053. URL https://www.sciencedirect.com/science/article/pii/S0377221717302758.

David Füßler and Nils Boysen. Efficient order processing in an inverse order picking system. *Computers Operations Research*, 88, 07 2017. doi: 10.1016/j.cor.2017.07.005.

Cristiano Arbex Valle and John E Beasley. Order allocation, rack allocation and rack sequencing for pickers in a mobile rack environment. *Computers Operations Research*, 125:105090, 2021a. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2020.105090. URL https://www.sciencedirect.com/science/article/pii/S0305054820302070.

R.D. Meller, D. Nazzal, and L.M. Thomas. Collaborative bots in distribution centers. *Proceedings of the 15th IMHRC (Savannah, Georgia. USA 2018)*, page 1 – 7, 2018. URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85092737915&partnerID=40&md5=659abd7c967aa5013af5983008db415b.

Zabih Ghelichi and Srikanth Kilaru. Analytical models for collaborative autonomous mobile robot solutions in fulfillment centers. *Applied Mathematical Modelling*, 91:438–457, 2021. ISSN 0307-904X. doi: https://doi.org/10.1016/j.apm.2020.09.059. URL https://www.sciencedirect.com/science/article/pii/S0307904X20305801.

Patrick Kübler, Christoph H. Glock, and Thomas Bauernhansl. A new iterative method for solving the joint dynamic storage location assignment, order batching and picker routing problem in manual picker-to-parts warehouses. *Computers Industrial Engineering*, 147:106645, 2020. ISSN 0360-8352. doi: https://doi.org/10.1016/j.cie.2020.106645. URL https://www.sciencedirect.com/science/article/pii/S036083522030379X.

Cristiano Arbex Valle and John E Beasley. Order batching using an approximation for the distance travelled by pickers. *European Journal of Operational Research*, 284(2):

460–484, 2020. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2020.01.022. URL https://www.sciencedirect.com/science/article/pii/S0377221720300436.

Babiche Aerts, Trijntje Cornelissens, and Kenneth Sörensen. The joint order batching and picker routing problem: Modelled and solved as a clustered vehicle routing problem. *Computers Operations Research*, 129:105168, 2021. ISSN 0305-0548. doi: https://doi.org/10.1016/j.cor.2020.105168. URL https://www.sciencedirect.com/science/article/pii/S0305054820302859.

Çağla Cergibozan and A. Tasan. Genetic algorithm based approaches to solve the order batching problem and a case study in a distribution center. *Journal of Intelligent Manufacturing*, 33:1–13, 01 2022. doi: 10.1007/s10845-020-01653-3.

Marek Matusiak, René de Koster, Leo Kroon, and Jari Saarinen. A fast simulated annealing method for batching precedence-constrained customer orders in a warehouse. *European Journal of Operational Research*, 236(3):968–977, 2014. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2013.06.001. URL https://www.sciencedirect.com/science/article/pii/S0377221713004670. Vehicle Routing and Distribution Logistics.

Chih-Ming Hsu, Kai-Ying Chen, and Mu-Chen Chen. Batching orders in warehouses by minimizing travel distance with genetic algorithms. *Computers in Industry*, 56:169–178, 02 2005. doi: 10.1016/j.compind.2004.06.001.

Thomas Lienert, Tobias Staab, C Fottner Ludwig, et al. Simulation-based performance analysis in robotic mobile fulfilment systems. In *Proceedings of the 8th International Conference on Simulation and Modeling Methodologies, Technologies and Applications*, 2018.

Yeming Gong, Mingzhou Jin, and Zhe Yuan. Robotic mobile fulfilment systems considering customer classes. *International Journal of Production Research*, pages 1–18, 12 2020. doi: 10.1080/00207543.2020.1779370.

Kun Wang, Yiming Yang, and Ruixue Li. Travel time models for the rack-moving mobile robot system. *International Journal of Production Research*, 58:1–19, 08 2019. doi: 10.1080/00207543.2019.1652778.

Felix Weidinger, Nils Boysen, and Dirk Briskorn. Storage assignment with rack-moving mobile robots in kiva warehouses. *Transportation Science*, 52, 11 2018. doi: 10.1287/trsc.2018.0826.

Rong Yuan, Tolga Cezik, and Stephen Graves. Velocity-based storage assignment in semi-automated storage systems. *SSRN Electronic Journal*, 01 2016. doi: 10.2139/ssrn.2889354.

Tim Lamballais Tessensohn, Debjit Roy, and René BM De Koster. Inventory allocation in robotic mobile fulfillment systems. *IISE Transactions*, 52(1):1–17, 2020.

Zheng Wang, Jiuh-Biing Sheu, and Chung Teo. Robot scheduling for mobile-rack warehouses: Human-robot coordinated order picking systems. *Production and Operations Management*, 03 2021. doi: 10.1111/poms.13406.

Marius Merschformann, Tim Lamballais, and René De Koster. Decision rules for robotic mobile fulfillment systems. *Operations Research Perspectives*, 6, 01 2018. doi: 10.1016/j.orp.2019.100128.

Yanling Zhuang, Yun Zhou, Yufei Yuan, Xiangpei Hu, and Elkafi Hassini. Order picking optimization with rack-moving mobile robots and multiple workstations. *European Journal of Operational Research*, 300(2):527–544, 2022. ISSN 0377-2217. doi:

https://doi.org/10.1016/j.ejor.2021.08.003. URL https://www.sciencedirect.com/science/article/pii/S0377221721006706.

Cristiano Arbex Valle and John E Beasley. A two-stage approach for order and rack allocation in a mobile rack environment. *arXiv preprint arXiv:2110.11550*, 2021b.

Zhi Li, Ali Vatankhah Barenji, Jiazhi Jiang, Ray Zhong, and Gangyan Xu. A mechanism for scheduling multi robot intelligent warehouse system face with dynamic demand. *Journal of Intelligent Manufacturing*, 31, 02 2020b. doi: 10.1007/s10845-018-1459-y.

Lu Zhen and Haolin Li. A literature review of smart warehouse operations management. *Frontiers of Engineering Management*, 9, 01 2022. doi: 10.1007/s42524-021-0178-9.

Ítalo Barros and Tiago Nascimento. Robotic mobile fulfillment systems: A survey on recent developments and research opportunities. *Robotics and Autonomous Systems*, 137, 01 2021. doi: 10.1016/j.robot.2021.103729.

Kaveh Azadeh, René De Koster, and Debjit Roy. Robotized warehouse systems: Developments and research opportunities. *SSRN Electronic Journal*, 01 2017. doi: 10.2139/ssrn.2977779.

René de Koster, Tho Le-Duc, and Kees Jan Roodbergen. Design and control of warehouse order picking: A literature review. *European Journal of Operational Research*, 182(2): 481–501, 2007. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2006.07.009. URL https://www.sciencedirect.com/science/article/pii/S0377221706006473.

Jun-tao Li. Design optimization of amazon robotics. *Automation, Control and Intelligent Systems*, 4:48, 01 2016. doi: 10.11648/j.acis.20160402.17.

Antonios Chatzisavvas, Petros Chatzitoulousis, Dimitris Ziouzios, and Minas Dasygenis. A routing and task-allocation algorithm for robotic groups in warehouse environments. *Information*, 13(6), 2022. ISSN 2078-2489. doi: 10.3390/info13060288. URL https://www.mdpi.com/2078-2489/13/6/288.

Maojia P. Li, Prashant Sankaran, Michael E. Kuhl, Raymond Ptucha, Amlan Ganguly, and Andres Kwasinski. Task selection by autonomous mobile robots in a warehouse using deep reinforcement learning. In *Proceedings of the Winter Simulation Conference*, WSC '19, page 680–688. IEEE Press, 2020c. ISBN 9781728132839.

Yi Li, Ruining Zhang, and Dandan Jiang. Order-picking efficiency in e-commerce warehouses: A literature review. *Journal of Theoretical and Applied Electronic Commerce Research*, 17:1812–1830, 12 2022. doi: 10.3390/jtaer17040091.

Chi-Guhn Lee and Zhong Ma. The generalized quadratic assignment problem. 01 2004.